B O L T   B E R A N E K   A N D   N E W M A N   I N C

C O N S U L T I N G · D E V E L O P M E N T · R E S E A R C H

BBN Report No. 3240

12

SPEECH UNDERSTANDING SYSTEMS

Quarterly Technical Progress Report No. 5

30 October 1975 to 31 January 1976

Sponsored by
Advanced Research Projects Agency
ARPA Order No. 2904

This research was supported by the Advanced Research Projects Agency of the Department of Defense and was monitored by ONR under Contract No. N00014-75-C-0533.

#-4- AD-A018 683

ERRATUM

With regard to the enclosed report, BBN Report No. 3240, SPEECH UNDERSTANDING SYSTEMS, Quarterly Technical Progress Report No. 5, the following should be noted:

Page 78. The equation as shown is incorrect and should read as follows:

$$d = \log \frac{\sum_{i=-p}^{p} b_i' R_i}{\sum_{i=-q}^{q} b_i R_i}$$

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER (14) BBN ~~Report No.~~-3240	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) (6) SPEECH UNDERSTANDING SYSTEMS. (9) Quarterly ~~Technical~~ Progress Report, No. 5 30 Oct~~ober 1975 to~~ 31 Jan~~uary 1976~~		5. TYPE OF REPORT & PERIOD COVERED Quarterly Tech. Prog. Report 30 Oct. 1975 to 31 Jan. 1976
		6. PERFORMING ORG. REPORT NUMBER BBN Report No. 3240
7. AUTHOR(s) (10) ~~W.~~/Woods, M./Bates, G./Brown, B./Bruce, C./Cook L. Gould, J. Klovstad, J. Makhoul, B. Nash-Webber, R. Schwartz, J. Wolf, V. Zue		8. CONTRACT OR GRANT NUMBER(s) (15) N00014-75-C-0533, ARPA Order-2904
9. PERFORMING ORGANIZATION NAME AND ADDRESS Bolt Beranek and Newman Inc. 50 Moulton Street Cambridge, MA 02138		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 5D30
11. CONTROLLING OFFICE NAME AND ADDRESS ONR Department of the Navy Arlington, VA 22217		12. REPORT DATE (11) Feb~~ruary~~ 1976
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) (12) 107p.		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION DOWNGRADING SCHEDULE

William A.

16. DISTRIBUTION STATEMENT (of this Report)

Distribution of this document is unlimited. It may be released to the Clearinghouse, Department of Commerce for sale to the general public.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Acoustic-Phonetic Experimental Facility, Acoustic-Phonetics, Acoustic-Phonetic Recognition, Acoustic-Phonetic Rules, Acoustics, Allophones, Cepstrum, Dictionary Expansion, Dynamic Programming, Frequency Warping, Interactive Programming, Inverse Filtering, Labeling, Lattice Filtering, Lexical Retrieval, Linear Prediction, Multi-component Systems, Natural

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This

This report describes recent progress of the BBN speech understanding system project covering the period from November 1975 to January 1976. The BBN speech understanding project is an effort to develop a continuous speech understanding system which uses syntactic, semantic and pragmatic support from higher level linguistic knowledge sources to compensate for the inherent acoustic indeterminacies in continuous spoken utterances. These knowledge sources are integrated with sophisticated signal processing

(Cont'd)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

060100

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

19. Key Words - cont'd.

Language Retrieval System, Natural Language Understanding, Parsing, Phonological Rules, Phonology, Probabilistic Labeling, Question-Answering, Recognition Strategies, Resource Allocation, Segmentation, Segment Lattice, Semantic Grammar, Spectral Warping, Speech, Speech Data Base, Speech Processing, Speech Recognition, Speech Understanding, SUR, Synthesis-By-Rule, System Organization, Word Recognition, Word Spotting, Word Verification.

20. Abstract - cont'd. (P 1473 A)

and acoustic-phonetic analysis of the input signal, to produce a total system for understanding continuous speech. The system contains components for signal analysis, acoustic parameter extraction, acoustic-phonetic analysis of the signal, phonological expansion of the lexicon, lexical matching and retrieval, syntactic analysis and prediction, semantic analysis and prediction, pragmatic evaluation and prediction, and inferential fact retrieval and question answering, as well as synthesized text or spoken output.

1473 P

A

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

SPEECH UNDERSTANDING SYSTEMS
Quarterly Technical Progress Report No. 5
30 October 1975 to 31 January 1976

ARPA Order No. 2904

Program Code No. 5D30

Name of Contractor:
Bolt Beranek and Newman Inc.

Effective Date of Contract:
30 October 1974

Contract Expiration Date:
29 October 1976*

Amount of Contract: $1,041,261

Contract No. N00014-75-C-0533

Principal Investigator:
William A. Woods
(617) 491 1850 x361

Scientific Officer:
Marvin Denicoff

Title:
SPEECH UNDERSTANDING SYSTEMS

QTPR Editor:
Bonnie Nash-Webber
(617) 491-1850 x227

Sponsored by
Advanced Research Projects Agency
ARPA Order No. 2904

*Modification is in process extending period of performance to 29 October 1976.

This research was supported by the Advanced Research Projects Agency of the Department of Defense and was monitored by ONR under Contract No. N00014-75-C-0533.

Table of Contents

I. PROGRESS OVERVIEWS

A. Acoustic-Phonetic Recognition

Several improvements were made to existing procedures in the APR. The sonorant-obstruent decision was improved by modifying control parameters in the dip detector. Acoustic-phonetic rules were added to detect sentence initial [HH] and [Q] (glottal stop), sentence initial plosives, and sentence final unreleased plosives. In addition a rule was introduced which uses formant transitions to separate [M] from [N]. This rule makes the correct decision 80% of the time. We believe that an identical algorithm can be used to distinguish other labial consonants from their dental counterparts.

This quarter, we also experimented with the use of context dependent phonemes (allophones), by defining 5 allophones of [T] and 4 allophones of [IH]. This was found to improve performance.

Finally, programs were written which produce quick printer displays for easy debugging and for evaluation of performance. We also experimented with different procedures for gathering the matrix of confusions between phonemes and segment labels. We found that our automatic procedure for gathering these statistics did not always perform as well as a human would. The quick segment lattice displays allow human intervention with a minimum of effort.

B. Phonology Dictionary Expansion

During the last quarter the set of lexical items in our dictionary, their phonetic spellings, and our collection of rules have stabilized to the extent that we are able to make meaningful measurements of expansion ratios at various stages of the dictionary expansion.

Initially a dictionary consists of a set of roots. In a preliminary expansion, roots that inflect regularly are used to generate their inflected forms. The resulting set of uninflected roots, irregular roots and inflected forms are termed words. Each word may have more than one pronunciation baseform. In a second expansion, a collection of phonological rules is used to generate from these baseforms a set of pronunciations we call pronunciation-1. The final expansion uses a set of rules reflecting either allophonic variations or APR dependency and produces a set of final pronunciations which we call pronunciation-2.

The current dictionary has 507 roots which expand into 702 words. The total number of baseforms is 818. These baseforms expand into 1,562 pronunciation variants due to phonological effects. The final number of pronunciations is 1,910.

Table I summarizes the expansion in terms of the intermediate and final expansion ratios. For example, the cell at the junction of the 2nd row and the 3rd column indicates that the dictionary goes through a 17% expansion due to multiple baseforms of certain words.

Table II gives expansion ratios of a larger (1,072 roots) dictionary which will eventually be used in the system. Comparison of the two tables indicates that doubling the size of lexicon produces roughly the same expansion ratios, thus giving us assurance of the relative stability of our expansion processes.

CURRENT DICTIONARY (507 ROOTS)

	R	W	B	P1	P2
R	1.0	1.38	1.61	3.08	3.77
W		1.0	1.17	2.23	2.72
B			1.0	1.91	2.33
P1				1.0	1.22

TABLE I

LARGER DICTIONARY (1072 ROOTS)

	R	W	B	P1	P2
R	1.0	1.34	1.51	3.25	4.00
W		1.0	1.13	2.43	2.99
B			1.0	2.16	2.65
P1				1.0	1.23

TABLE II

C. Lexical Retrieval

During the past quarter many small program changes were made to the lexical retrieval component of which the following few are most significant. We introduced the notion of "pronunciation features" to characterize a particular pronunciation of a given word, for example REDUCED, UNVOICED, STRIFRIC (i.e., strident fricative), MODFORM (i.e., modifier form), and HEAD. While many of these features had previously been handled by separate lexical items in the dictionary, for example "THE-R" (THE with the feature REDUCED), they are now all handled by associating with each word match a list of appropriate features. This has the advantage for the higher level components of reducing the number of words they need to know about. Though these pronunciation features can not be used to restrict the set of words sought in a proposal to the lexical retrieval component, they can be used as additional restrictions on the results of the proposal. For example, Syntax can now propose THAT (as a definite article) and filter the results of that proposal for matches which do not have the feature REDUCED, since that feature would only be acceptable for uses of THAT as a relative pronoun.

A program which produces exhaustive word boundary rules from a simply stated representation was also written, thereby insuring a complete enumeration of each specified rule. In addition, the way in which log probability ratios are calculated was changed in order to eliminate a roundoff error otherwise introduced for small numbers. To facilitate our evaluation of the acoustic phonetic component, a program was written to print out in matrix format the contents of the scoring matrix. Another recently written program automatically collects the statistics from a manually prepared dictionary entry - segment lattice alignment and produces a pseudo hand label file (i.e., like an "ideal" hand label file in every respect except origin). Since these alignments are more closely related to actually attainable alignments the resulting statistics are more suitable than those obtained by using an "ideal" hand label file.

D. Syntax

1. Parser

During the past quarter a new syntactic parser was designed and implemented to take greater advantage of the predictive information contained in our ATN grammars. With the new parser, one can efficiently parse and make prediction across any number of uncompleted levels of an ATN grammar.

The primary reason for investing effort in a new parser at this time was that we found, in running the system on many utterances, that we needed a parser which could find all possible paths through an island of word matches and then predict all words and categories which could occur adjacent to that island according to the grammar. The new parser fulfills these needs by keeping track of all possible paths through each island that it has processed.

To do this, we have introduced three different types of "configurations". A segment configuration (SCONFIG) represents one path (i.e., sequence of arcs) that can be taken through (some part of) an island at one level of the grammar. An SCONFIG records the relevant register settings, the beginning and end state of the path, the part of the island consumed there, and any arc actions which have not yet been performed. A path configuration (PCONFIG) is a list of SCONFIGS which form one continuous path through an island. An island configuration (ICONFIG) is a list of all possible paths (PCONFIGS) through a particular island.

In the new parser, once an initial ICONFIG has been set up for a one-word island, parsing proceeds by successively adding words to its ends. When the parser has just constructed a one-word island or has constructed an island by adding a word, it makes proposals for all possible words and categories which can be combined with that island at either end. By using details of the recursion history in the PCONFIGs and precompiled information about the accessibility of states from other ones via non-consuming arcs (PUSH, JUMP, and POP arcs), the set of proposals is restricted as tightly as possible, taking into account not only the states at the end of the island, but also possible sequences of PUSHes within the island. The addition of words to one end of the island may restrict the proposals at its other end by removing some of the possible paths through the island.

Each proposal remembers the particular arc of the grammar that was used to make the proposal. When the parser is later given an event that results from a proposal, it invokes a search routine to discover all sequences of non-consuming arcs that can be used to connect the proposing arc to some state at the end of the island. Thus, the enumeration of the various possible sequences of PUSH, JUMP, and POP arcs is not done at the ends of islands prior to proposals, but rather in between an island and a proposed word after the proposal has been found.

Register setting and testing actions are done at the earliest possible moment as determined by scope statements in the grammar. Whenever one of the endpoints of an island reaches the beginning or end of the utterance, the parser checks for sequences of non-consuming arcs which will connect that end of the island to the initial or some final state of the grammar (as appropriate), in order to eliminate paths that cannot be compatible with a complete parse.

The effect of these changes is that the parser can efficiently make specific proposals for all possible words and categorics adjacent to either end of an island, taking into account what is happening at the other end of the island and all of the usable register contents and recursion history within the island.

Prior to implementing the new parser, work was done on the old parser to implement many of the changes described in the paragraph on the new parser. The experience gained in making and testing the new ideas on the old parser as an invaluable aid in designing and implementing the new one.

2. Grammar

The grammar (SMALLGRAM) has been enlarged, tested, debugged, and documented this quarter. We have added a facility for parsing fourteen types of utterances that are not complete sentences but rather phrases that may be meaningful in certain dialogue contexts. Examples of such phrases are amounts of money, numbers (perhaps expressed as digit strings), dates (including prepositional phrases that are date modifiers), names of people, geographic locations, institutions (e.g., SCRL, Carnegie-Mellon), budgets, budget items, projects, contracts, meetings, and short commands and answers (e.g., yes, no, stop, go on, I don't know).

The arcs in the grammar which allow these short utterances to be accepted as complete utterances may be taken if and only if a flag has been set (by the retrieval component) which indicates the type of utterance expected. When this flag is set, an entire sentence as well as a short utterance of the specified type is acceptable. We intend to experiment with "turning off" portions of the grammar based on these predictions of likely utterance types.

The grammar has been extended to allow (and hence predict) pauses in a limited number of specific contexts (for example, at the beginning and end of a sentence and after the "and" of a dollars-and-cents expression). We have included paths in the grammar to process questions and commands about projects, contracts, budgets and budget items, charging amounts or trips to budgets or budget items, and budgeting, scheduling, or canceling trips. The parse trees for a number of constituents (e.g., dates and some noun phrases) have been modified to conform to the requirements of the retrieval and semantic components which will process the completed parse trees.

We have also documented the grammar by producing both an accurate picture of the context-free aspects of the grammar and a semi-BNF form of the grammar which indicates much more clearly than the grammar listing what sorts of sentences are accepted by the grammar. From this BNF form, we made up a list of 130 sentences which will be used to exhaustively test and further debug SMALLGRAM.

E. <u>The Travel Budget Manager's Assistant</u>

The data base organization for the travel budget manager's assistant has been generalized somewhat during the last quarter to accommodate a more general budget management problem. In particular, the notion of "budget item" has been isolated as the basic combining element for a budget regardless of the purpose of the budget. A budget item represents an allocation of resources needed to carry out a plan and covers various activities pertaining to that plan.

For the current travel data base the "allocation" of a budget item contains information about money initially and currently allocated, money spent and money remaining. The "plan" for a budget item is a (perhaps vague) description of a trip or trips, such as "3 people to the ASA Meeting". The budget item then "covers" the actual trips which the manager associates with the plan.

The new data base structures permit a distinction between "proposed trips" and "travel plans". In the first case the structure represents an actual trip, with a specific traveler, trip number, dates, etc., which has not yet been taken. In the second case the structure represents an expectation of the manager about a class of trips, usually without dates, and often specified only in terms of a number of travelers and a purpose and/or destination. Using real data from the BBN Speech Understanding project and a few related projects, we have found this to be a useful distinction.

The current data base has 26 actual, taken trips and 25 budget items. (The nearness of these numbers is coincidental since some budget items cover several trips while many cover none.) There are about 1400 nodes in the semantic network now (including those used by the Semantic Recognizer).

We plan to continue maintaining the data base as trips are taken and travel budget plans change. Work on the data base accessing functions will also continue particularly in the direction of increasing efficiency. This will require work both at the level of SEMNET (basic semantic network) functions which perform the actual data base storage and retrieval and at the level of the command language where more intelligent structuring of the command can produce dramatic speedups in the search. In the latter area, for example, the principal search command, FIND:, checks its pattern description first to see if any element of the pattern necessarily refers to a singleton set. If so, then search is organized around that item rather than in a blind enumerative fashion.

F. Verification

During the past quarter, we have integrated the verification component into the rest of the speech understanding system. Since then, we have been exploring various ways of using verification within the overall control strategy. The most successful technique we have found so far is to use verification as a thresholding mechanism for evaluating new events (i.e., new words in the context of a given theory). Using this technique, we have been able to remove from consideration many incorrect events which would have otherwise required syntactic processing. However it has sometimes been the case that a correct event has failed to pass the verification threshold thus removing it from consideration. We are currently investigating appropriate modifications to the use of verification within the overall control strategy.

We have also implemented a facility for gathering statistics of verification's performance during the understanding process. Briefly, the component writes the results of each verify request (phonetic spelling, match score and word alignment) onto a disk file. These files yield the desired statistics in addition to providing a useful check on the operation of other components in the system.

A new version of the synthesis-by-rule program was developed during the past quarter for verification purposes and tested extensively in our audio response component. The program has now been integrated into the verification component. Currently, we are working to improve the performance of verification on two fronts. We are trying some new synthesis experiments to improve the parametric model generated by the synthesis-by-rule program. In terms of computation time, we are attempting to improve the component's performance by rewriting its dynamic programming algorithm in PDP-10 assembler language.

G. System Organization and Recognition Strategies

During the past quarter, the remaining fork interfaces between the system's components were implemented and debugged, and a fully automatic, almost daily cycling of the system on new utterances was begun. Both our newly acquired ability to call the verification component and the increased predictive power of the syntactic component led to modifications of our main recognition strategy, now called an "island driven strategy" (See II.F.), from that reported in the last QPR. At the end of the quarter, the primary characteristics of this strategy were:

1. The use of Syntax to predict all words and classes on each side of the island it was currently considering. This replaced the "priming the pump" procedure that had the Lexical Retrieval component making anchored scans on each side of the island for the best matching words.

2. The use of the Verification component to score events and remove those which did not meet a threshold. Most often, this eliminated events which would otherwise have been sent to Syntax, at the time, a much more expensive process, but it brought in the possibility for a correct event to be eliminated, which was disastrous. However, in the absence of a relative evaluation of the scores from lexical retrieval and verification, this seemed the best way of using the verification component. In the coming quarter, we will be working on making verification scores commensurate with those of lexical retrieval in order to use both scores in computing the primary evaluation and ranking of events.

3. The shelving of events for a father theory while a son theory was being processed, thus making this strategy strongly depth first. This was done because of Syntax's early inability to process many theories before running out of space. The new parser brought up near the end of the quarter did not have this problem. Next quarter, we will be expanding the amount of parallelism carried on.

As new versions of segment lattices were produced during the quarter, this island-driven strategy was run on a core of 12 utterances, as well as ones newly acquired through cycling the complete system automatically. Approximately one-third of the core utterances were recognized consistently. Our performance on new utterances was much lower due to several previously untreated acoustic phenomena in the new utterances. The purpose of running these new utterances was to discover such untreated phenomena and develop capabilities for dealing with them. Most of the problems uncovered were either simple to correct or involved phenomena which we had known about qualitatively and required concrete examples in order to decide which methods to use for dealing with them. Our major problems have to do with surviving acoustic segmentation errors at word boundaries, and we expect a technique which we have already partially coded to solve a large number of these problems in the system.

Table III gives timing summary results for 7 of the utterances which were recognized in the last major run of the quarter. (Note: the first three examples were using the old parser, while the last four were done with the new parser, which is faster.) These timing data are incomplete and are not necessarily representative, but are the best data available at the present time.

UTTN	Time for Initial Scan	Duration & # of Segments	Word Verification Calls	Time/ Call	Syntactic Event Processing	Time/ Call	Proposals W = Words C = Classes	Proposal Calls	Proposals /Calls	Time/ Call
JJW100 N	122.579	3.35 sec 49	85	5.742	25	27.060	268 W 22 C	24	11.1 W	3.395
JJW102 N	83.652	1.45 sec 28	30	5.464	5	26.852	87 W 2 C	5	17.4 W .4 C	3.739
JJW104 N	59.515	1.85 sec 22	30	6.519	4	25.327	99 W 20 C	4	25.0 W 5.0 C	4.360
JJW110 N	59.025	1.30 sec 21	26	5.306	4	14.006	67 W 1 C	4	17.0 W .25 C	3.450
JJW110 O	70.580	1.30 sec 21	28	5.994	4	15.516	76 W 2 C	4	19.0 W .5 C	4.089
JJW108 O	93.453	1.75 sec 29	22	6.386	6	12.860	48 W 11 C	6	8.0 W 1.9 C	3.820
JJW101 O	96.957	2.20 sec 36	22	6.706	8	8.961	42 W 13 C	8	5.25 W 1.6 C	3.911

(Time given in CPU seconds)

TABLE III

II. TECHNICAL NOTES

A. New Lattice Methods for Linear Prediction

John Makhoul

This paper presents a new formulation for linear prediction, which we call the covariance lattice method. The method is viewed as one of a class of lattice methods which guarantee the stability of the all-pole filter, with or without windowing of the signal, with finite wordlength computations, and with the number of computations being comparable to the traditional autocorrelation and covariance methods. In addition, quantization of the reflection coefficients can be accomplished within the recursion for retention of accuracy in representation.

1. Introduction

The autocorrelation method of linear prediction [1] guarantees the stability of the all-pole filter, but has the disadvantage that windowing of the signal causes some unwanted distortion in the spectrum. In practice, even the stability is not always guaranteed with finite wordlength (FWL) computations [2]. On the other hand, the covariance method does not guarantee the stability of the filter, even with floating point computation, but has the advantage that there is no windowing of the signal. One solution to these problems was given by Itakura [3] in his lattice formulation. In this method, filter stability is guaranteed, with no windowing, and with FWL computations. Unfortunately, this is accomplished with about a four-fold increase in computation over the other two methods.

This paper presents a class of lattice methods which have all the properties of a regular lattice but where the number of computations is comparable to the autocorrelation and covariance methods. In these methods the "forward" and "backward" residuals are not computed. The reflection coefficients are computed directly from the covariance of the input signal.

2. Lattice Formulations

In linear prediction, the signal spectrum is modelled by an all-pole spectrum with a transfer function given by

$$H(z) = \frac{G}{A(z)}, \tag{1}$$

where $$A(z) = \sum_{k=0}^{p} a_k z^{-k}, \; a_0 = 1, \tag{2}$$

is known as the inverse filter, G is a gain factor, a_k are the predictor coefficients, and p is the number of poles or predictor coefficients in the model. If H(z) is stable, A(z) can be implemented as a lattice filter, as shown in Fig. 1. The reflection (or partial correlation) coefficients K_i in the lattice are uniquely related to the predictor coefficients. Given K_i, $1 \le i \le p$, the set $\{a_k\}$ is computed by the recursive relation:

$$a_i^{(i)} = K_i$$

$$a_j^{(i)} = a_j^{(i-1)} + K_i \, a_{i-j}^{(i-1)} , \quad 1 \le j \le i-1, \tag{3}$$

where the equations in (3) are computed recursively for $i=1,2,\ldots,p$. The final solution is given by $a_j=a_j^{(p)}$, $1 \le j \le p$. For a stable H(z), one must have:

$$|K_i| < 1, \; 1 \le i \le p \; . \tag{4}$$

In the lattice formulation, the reflection coefficients can be computed by minimizing some error norm of the forward residual $f_m(n)$ or the backward residual $b_m(n)$, or a combination of the two. From Fig. 1, the following relations hold:

$$f_0(n) = b_0(n) = s(n) \; , \tag{5a}$$

$$f_{m+1}(n) = f_m(n) + K_{m+1} \, b_m(n-1) \; , \tag{5b}$$

$$b_{m+1}(n) = K_{m+1} \, f_m(n) + b_m(n-1) \; . \tag{5c}$$

s(n) is the input signal and $e(n)=f_p(n)$ is the output residual.

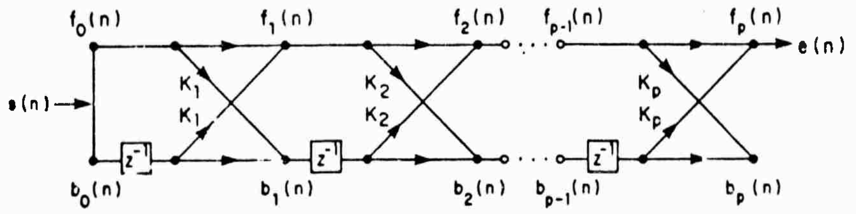

Fig. 1. Lattice inverse filter.

We shall give several methods for the determination of the reflection coefficients. These methods depend on different ways of correlating the forward and backward residuals. Below, we shall make use of the following definitions:

$$F_m(n) = E[f_m^2(n)] \tag{6a}$$

$$B_m(n) = E[b_m^2(n)] \tag{6b}$$

$$C_m(n) = E[f_m(n)b_m(n-1)] \ , \tag{6c}$$

where $E(.)$ denotes expected value. The left hand side of each of the equations in (6) is a function of n because we are making the general assumption that the signals are nonstationary. (Subscripts, etc., will be dropped sometimes for convenience.)

(a) Forward Method

In this method the reflection coefficient at stage m+1 is obtained as a result of the minimization of an error norm given by the variance (or mean square) of the forward residual:

$$F_{m+1}(n) = E[f_{m+1}^2(n)] \ . \tag{7}$$

By substituting (5b) in (7) and differentiating with respect to Km+1, one obtains:

$$K^{f}_{m+1} = - \frac{E[f_m(n) b_m(n-1)]}{E[b^2_m(n-1)]} = - \frac{C_m(n)}{B_m(n-1)} \tag{8}$$

This method of computing the filter parameters is similar to the autocorrelation and covariance methods in that the mean squared forward error is minimized.

(b) Backward Method

In this case, the minimization is performed on the variance of the backward residual at stage m+1. From (5c) and (6b), the minimization of $B_{m+1}(n)$ leads to:

$$K^{b}_{m+1} = - \frac{E[f_m(n) b_m(n-1)]}{E[f^2_m(n)]} = - \frac{C_m(n)}{F_m(n)} \quad . \tag{9}$$

Note that, since $F_m(n)$ and $B_m(n-1)$ are both nonnegative and the numerators in (8) and (9) are identical, K^f and K^b always have the same sign S:

$$S = \text{sign } K^f = \text{sign } K^b \quad . \tag{10}$$

(c) Geometric Mean Method (Itakura)

The main problem in the above two techniques is that the computed reflection coefficients are not always guaranteed to be less than 1 in magnitude, i.e., the stability of H(z) is not guaranteed. One solution to this problem was offered by Itakura [3] where the reflection coefficients are computed from

$$K^I_{m+1} = -\frac{E[f_m(n)b_m(n-1)]}{\sqrt{E[f^2_m(n)]E[b^2_m(n-1)]}} = -\frac{C_m(n)}{\sqrt{F_m(n)B_m(n-1)}} . \tag{11}$$

K^I_{m+1} is the negative of the statistical correlation between $f_m(n)$ and $b_m(n-1)$; hence, property (4) follows. To the author's knowledge, (11) cannot be derived directly by minimizing some error criterion. However, from (8), (9) and (11), one can easily show that K^I is the geometric mean of K^f and K^b:

$$K^I = S\sqrt{K^f K^b} \tag{12}$$

where S is given by (10). From the properties of the geometric mean, it follows that:

$$\min[|K^f|,|K^b|] \le |K^I| \le \max[|K^f|,|K^b|] . \tag{13}$$

Now, since $|K^I|<1$, it follows that <u>if the magnitude of</u>

either K^f or K^b is greater than 1, the magnitude of the other is necessarily less than 1. This leads us to another definition for the reflection coefficient.

(d) Minimum Method

$$K^M = S \min[|K^f|, |K^b|] \quad . \tag{14}$$

This says that, at each stage, compute K^f and K^b and choose as the reflection coefficient the one with the smaller magnitude.

(e) General Method

Between K^M and K^I there are an infinity of values that can be chosen as valid reflection coefficients (i.e., $|K|<1$). These can be conveniently defined by taking the generalized rth mean of K^f and K^b:

$$K^r = S\left[\frac{1}{2}(|K^f|^r + |K^b|^r)\right]^{1/r} \quad . \tag{15}$$

As $r \to 0$, $K^r \to K^I$, the geometric mean. For $r>0$, K^r cannot be guaranteed to satisfy (4). Therefore, for K^r to be a reflection coefficient, we must have $r \leq 0$. In particular:

$$K^0 = K^I, \; K^{-\infty} = K^M \quad . \tag{16}$$

If the signal is stationary, one can show that $K^f = K^b$, and that

$$K^r = K^f = K^b, \text{ all } r. \text{ (Stationary Case)} \tag{17}$$

(f) Harmonic Mean Method (Burg)

There is one value of r for which K^r has some interesting properties, and that is r=-1. K^{-1}, then, would be the harmonic mean of K^f and K^b :

$$K^B = K^{-1} = \frac{2K^f K^b}{K^f + K^b} = -\frac{2C_m(n)}{F_m(n)+B_m(n-1)} \quad . \tag{18}$$

One can show that

$$|K^M| \le |K^B| \le |K^I| \quad . \tag{19}$$

In fact, Itakura used K^B as an approximation to K^I in (11) to avoid computing the square root.

One important property of K^B that is not shared by K^I and K^M, is that K^B results directly from the minimization of an error criterion. The error is defined as the sum of the variances of the forward and backward residuals:

$$E_{m+1}(n) = F_{m+1}(n) + B_{m+1}(n) \quad . \tag{20}$$

Using (5) and (6), one can show that the minimization of (20) indeed leads to (18). One can also show that the forward and backward minimum errors at stage m+1 are related to those at stage m by the following:

$$F_{m+1}(n) = \left[1-(K^B_{m+1})^2\right] F_m(n) \tag{21a}$$

$$B_{m+1}(n) = \left[1-(K^B_{m+1})^2\right] B_m(n-1) \quad . \tag{21b}$$

This formulation is originally due to Burg [4]; it has been used recently by Boll [5] and Atal [6].

(g) Discussion

If the signal s(n) is stationary, all the methods described above give the same result. In general, the signal cannot be assumed to be stationary and the different methods will give different results. Which method to choose in a particular situation is not clear cut. We tend to prefer the use of K^B in (18) because it minimizes a reasonable and well defined error and guarantees stability simultaneously, even for a nonstationary signal.

3. The Covariance-Lattice Method

If linear predictive analysis is to be performed on a regular computer, the number of computations for the lattice methods given above far exceeds that of the autocorrelation and covariance methods (see the first row of Fig. 2). This is unfortunate since, otherwise, lattice methods have superior properties when compared to the autocorrelation and covariance methods (see Fig. 3). Below, we derive a new method, called the covariance-lattice method, which has all the advantages of a regular lattice, but with an efficiency comparable to the two non-lattice methods.

	AUTOCORRELATION METHOD	COVARIANCE METHOD	REGULAR LATTICE (WITH RESIDUALS)
TRADITIONAL METHODS	$pN + p^2$	$pN + \frac{1}{6}p^3 + \frac{3}{2}p^2$	5 pN
NEW LATTICE METHODS	$pN + \frac{1}{6}p^3 + \frac{3}{2}p^2$	$pN + \frac{1}{2}p^3 + 2p^2$	5 pN

Fig. 2. Computational cost for traditional as compared to new lattice methods.

LINEAR PREDICTION METHOD	ADVANTAGES	DISADVANTAGES
AUTOCORRELATION	1. THEORETICAL STABILITY 2. COMPUTATIONALLY EFFICIENT	1. WINDOWING 2. POSSIBLE INSTABILITY WITH FWL COMPUTATION
COVARIANCE	1. NO WINDOWING 2. COMPUTATIONALLY EFFICIENT	1. STABILITY NOT GUARANTEED EVEN WITH FLOATING POINT
REGULAR LATTICE	1. WINDOWING NOT NECESSARY 2 STABILITY CAN BE GUARANTEED 3. NUMBER OF SAMPLES FOR ANALYSIS CAN BE REDUCED 4. REFLECTION COEFFICIENTS CAN BE QUANTIZED WITHIN RECURSION	1. COMPUTAT!ONALLY EXPENSIVE
COVARIANCE LATTICE	1-4. SAME AS FOR REGULAR LATTICE METHOD 5. COMPUTATIONALLY EFFICIENT	

Fig. 3. Comparison between different LP methods.

From the recursive relations in (3) and (5), one can show that

$$f_m(n) = \sum_{k=0}^{m} a_k^{(m)} s(n-k) \quad , \tag{22a}$$

$$b_m(n) = \sum_{k=0}^{m} a_k^{(m)} s(n-m+k) \quad . \tag{22b}$$

Squaring (22a) and taking the expected value, there results

$$F_m(n) = \sum_{k=0}^{m} \sum_{i=0}^{m} a_k^{(m)} a_i^{(m)} \phi(k,i) \quad , \tag{23}$$

where $\phi(k,i) = E\,[s(n-k)s(n-i)]$ (24)

is the nonstationary autocorrelation (or covariance) of the signal s(n). ($\phi(k,i)$ in (24) is technically a function of n, which has been dropped for convenience.) In a similar fashion one can show from (22b), with n replaced by n-1, that

$$B_m(n-1) = \sum_{k=0}^{m} \sum_{i=0}^{m} a_k^{(m)} a_i^{(m)} \phi(m+1-k,m+1-i), \tag{25}$$

$$C_m(n) = \sum_{k=0}^{m} \sum_{i=0}^{m} a_k^{(m)} a_i^{(m)} \phi(k,m+1-i) \quad . \tag{26}$$

Given the covariance of the signal, the reflection coefficient at stage m+1 can be computed from (23), (25) and (26) by substituting them in the desired formula for K_{m+1}. The name "covariance-lattice" stems from the fact that this is basically a lattice method that is computed from the covariance of the signal; it can be viewed as a way of stabilizing the covariance method. One salient feature is that the forward and backward residuals are never actually computed in this method. But this is not different from the non-lattice methods.

In the harmonic mean method (18), $F_m(n)$ need not be computed from (23); one can use (21a) instead, with m replaced by m-1. However, one must use (25) to compute $B_m(n-1)$; (21b) cannot be used because $B_{m-1}(n-2)$ would be needed and it is not readily available.

(a) Stationary Case

For a stationary signal, the covariance reduces to the autocorrelation:

$$\phi(k,i) = R(i-k) = R(k-i). \text{ (Stationary)} \tag{27}$$

From (23-27), it is clear that

$$F_m = B_m = \sum_{k=0}^{m} \sum_{i=0}^{m} a_k^{(m)} a_i^{(m)} R(i-k) , \tag{28}$$

$$\text{and } C_m = \sum_{k=0}^{m} \sum_{i=0}^{m} a_k^{(m)} a_i^{(m)} R(m+1-i-k) . \tag{29}$$

Making use of the normal equations [1]

$$\sum_{i=0}^{m} a_i^{(m)} R(i-k) = 0, \quad 1 \le k \le m , \tag{30}$$

and of (21), one can show that the stationary reflection coefficient is given by:

$$K_{m+1} = -\frac{C_m}{F_m} = -\frac{\sum_{k=0}^{m} a_k^{(m)} R(m+1-k)}{(1-K_m^2)F_{m-1}} \tag{31}$$

with $F_0=R_0$. (31) is exactly the equation used in the autocorrelation method.

(b) Quantization of Reflection Coefficients

One of the features of lattice methods is that the quantization of the reflection coefficients can be accomplished within the recursion, i.e., K_m can be quantized before K_{m+1} is computed. In this manner, it is hoped that some of the effects of quantization can be compensated for.

In applying the covariance-lattice procedure to the harmonic mean method, one must be careful to use (23) and not (21a) to compute $F_m(n)$. The reason is that (21a) is based on the optimality of K^B, which would no longer be true after quantization.

Similar reasoning can be applied to the autocorrelation method. Those who have tried to quantize K_m inside the recursion, have no doubt been met with serious difficulties. The reason is that (31) assumes the optimality of the predictor coefficients at stage m, which no longer would be true if K_m were quantized. The solution is to use (28) and (29), which make no assumptions of optimality. Thus, we have what we shall call the <u>autocorrelation lattice method</u>, where there is only one definition of K_{m+1}:

$$K_{m+1} = -\frac{C_m}{F_m}\,, \quad \text{(Autocorrelation-Lattice)} \tag{32}$$

where F_m and C_m are given by (28) and (29).

4. Computational Issues

(a) Simplifications

Equations (23),(25) and (26) can be rewritten to reduce the number of computations by about one half. The results for $C_m(n)$ and $F_m(n)+B_m(n-1)$ can be shown to be as follows:

$$\begin{aligned} C_m(n) = {} & \phi(0,m+1) + \sum_{k=1}^{m} a_k^{(m)}[\phi(0,m+1-k)+\phi(k,m+1)] \\ & + \sum_{k=1}^{m} [a_k^{(m)}]^2 \phi(k,m+1-k) \\ & + \sum_{k=1}^{m-1} \sum_{i=k+1}^{m} a_k^{(m)} a_i^{(m)} [\phi(k,m+1-i)+\phi(i,m+1-k)] \end{aligned} \tag{33}$$

$$\begin{aligned} F_m(n)+B_m(n-1) = {} & \phi(0,0)+\phi(m+1) \\ & + 2\sum_{k=1}^{m} a_k^{(m)}[\phi(0,k)+\phi(m+1,m+1-k)] \\ & + \sum_{k=1}^{m}[a_k^{(m)}]^2[\phi(k,k)+\phi(m+1-k,m+1-k)] \\ & + 2\sum_{k=1}^{m-1} \sum_{i=k+1}^{m} a_k^{(m)} a_i^{(m)} [\phi(k,i)+\phi(m+1-k,m+1-i)] \end{aligned} \tag{34}$$

(28) and (29) can also be simplified in a similar fashion.

(b) Covariance Computation

If the signal is known for $0 \leq n \leq N-1$, then one common method to compute the covariance is

$$\phi(k,i) = \sum_{n=p}^{N-1} s(n-k)s(n-i) \quad , \tag{35}$$

where p is the order of the desired predictor.

(c) Computational Cost

Fig. 2 shows a comparison of the number of computations for the different methods, where terms of order p have been neglected. The increase in computation for the covariance lattice method over non-lattice methods is not significant if N is large compared to p, which is usually the case. Furthermore, in the covariance lattice method, the number of signal samples can be reduced to about half that used in the autocorrelation method. This, not only reduces the number of computations, but also improves the spectral representation by reducing the amount of averaging.

5. Procedure

Below is the complete algorithm for what we believe currently tc be the best overall method for linear predictive analysis. It comprises the harmonic mean definition (18) for the reflection coefficients, and the covariance lattice method.

(a) Compute the covariances $\phi(k,i)$ for $k,i=0,1,\ldots,p$.

(b) $m \leftarrow 0$.

(c) Compute $C_m(n)$ and $F_m(n)+B_m(n-1)$ from (33) and (34), or from (23),(25) and (26).

(d) Compute K_{m+1} from (18).

(e) Quantize K_{m+1} if desired (perhaps using log area ratios [7] or some other technique).

(f) Using (3), compute the predictor coefficients $\{a_k^{(m+1)}\}$ from $\{a_k^{(m)}\}$ and K_{m+1}. Use the quantized value if K_{m+1} was quantized in (d).

(g) $m \leftarrow m+1$.

(h) If $m<p$, go to (c); otherwise exit.

References

[1] J. Makhoul, "Linear Prediction: A Tutorial Review," Proc. IEEE, 561-580, April 1975.

[2] J. Markel and A. Gray, Jr., "Fixed-Point Truncation Arithmetic Implementation of a Linear Prediction Autocorrelation Vocoder," IEEE Trans. ASSP, 273-281, 1974.

[3] F. Itakura and S. Saito, "Digital Filtering Techniques for Speech Analysis and Synthesis," 7th Int. Cong. Acoust., Budapest, 25-C-1, 1971.

[4] J. Burg, "A New Analysis Technique for Time Series Data," NATO Advanced Study Institute on Signal Processing, Enschede, Netherlands, 1968.

[5] S. Boll, "Selected Methods for Improving Synthesis Speech Quality Using Linear Predictive Coding: System Description, Coefficient Smoothing and STREAK," UTEC-CSc-74-151, Comp. Science Dept., Univ. Utah, 1974.

[6] B. Atal, M. Schroeder and V. Stover, "Voice-Excited

Predictive Coding System for Low Bit-Rate Transmission of Speech," Int. Conf. Comm., San Francisco, June 1975.

[7] R. Viswanathan and J. Makhoul, "Quantization Properties of Transmission Parameters in Linear Predictive Systems," IEEE Trans. ASSP, 309-321, June 1975.

B. Methods for Nonlinear Spectral Distortion of Speech Signals

John Makhoul

The spectral distortion of speech signals, without affecting the pitch or the speed of the signal, has met with some difficulty due to the need for pitch extraction. This paper presents a general analysis-synthesis scheme for the arbitrary spectral distortion of speech signals without the need for pitch extraction. Linear predictive warping, cepstral warping, and autocorrelation warping, are given as examples of the general scheme. Applications include the unscrambling of helium speech, spectral compression for the hard of hearing, bit rate reduction in speech compression systems, and efficiency of spectral representation for speech recognition systems.

1. Introduction

Arbitrary spectral distortion of any finite sampled signal can be easily accomplished by computing the discrete Fourier transform (DFT) of the signal, performing the desired spectral distortion, and then taking the inverse DFT. (The resulting signal is an approximation to the desired spectrally distorted signal in the same measure as the DFT is an approximation to the z transform. Arbitrary accuracy can be achieved by increasing the order of the DFT.) In applying this method to the spectral distortion of voiced speech signals, the spectral envelope is distorted as well as the voicing (pitch) characteristics. For many applications, the distortion is usually desired for the spectral envelope, but not for the pitch. Thus it becomes necessary to separate the pitch (source) information, distort the spectral envelope, and then resynthesize using the extracted source information.

Certain existing research systems [1-3] for the nonlinear spectral distortion of speech signals separate the source information by making voiced/unvoiced decisions and performing pitch extraction. A different approach was taken by Suzuki et al. [4] for the unscrambling of helium speech, where pitch extraction was not used. In their work, the source information was obtained as the residual signal in a linear predictive analysis of the speech signal. The spectral distortion was performed in the time domain on the impulse response of the all-pole filter. However, the only type of distortion attempted was a linear one, and it was effected by interpolation in the time domain. In this paper we describe a general analysis-synthesis system for the <u>nonlinear</u> spectral distortion of speech signals, <u>without the need for pitch extraction</u>. The generality of the system is achieved by performing the spectral distortion directly in the frequency domain. Three methods, linear predictive warping, cepstral warping, and autocorrelation warping, are given as examples of the general scheme.

2. General System

The general analysis-synthesis system for spectral distortion is shown in Fig. 1. The speech signal s(n) is passed through a filter whose magnitude frequency response is the inverse of the envelope of the signal spectrum. The output of the inverse filter is the residual signal e(n), which contains mainly the source information. Since all the resonant structure of the signal is removed by the inverse filter, e(n) will have an essentially flat spectral envelope. The residual signal is then used as input to a synthesis filter whose magnitude frequency response is equal to the desired distorted or warped spectral characteristics. The output of the synthesis filter, s′(n), is then the transformed signal with the same source characteristics as s(n), but with a spectrum that is a distorted version of the spectrum of s(n).

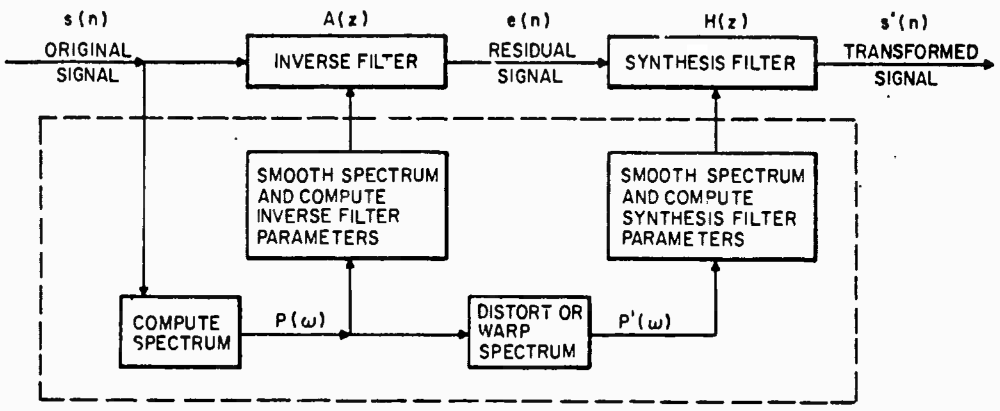

Fig. 1. General analysis-synthesis system for spectral warping.

One important property of the system in Fig. 1 is that the sampling rate remains fixed throughout the system. If for some reason the sampling rate at the output is desired to be different from that at the input, then one needs to perform down sampling on the residual signal, or else perform pitch extraction.

There remains the specification of the inverse filter and synthesis filter parameters. This is described next.

3. Nonparametric Warping

The dashed box in Fig. 1 shows the general scheme for spectral warping and for the specification of the inverse and synthesis filter parameters. A more detailed block diagram is shown in Fig. 2. The spectrum $P(\omega)$ of the signal $s(n)$ is computed by windowing the signal and taking the magnitude squared of its Fourier transform. $P(\omega)$ is then smoothed to retain the requisite resonant structure. The smoothed spectrum $\hat{P}(\omega)$ is then inverted. The resulting inverse smoothed spectrum is then used to determine the impulse response $a(n)$ of the inverse filter $A(z)$. Assuming a minimum phase implementation, $a(n)$ can be computed from $\hat{P}^{-1}(\omega)$ through the use of the cepstrum. Details can be found in Oppenheim and Schafer [5].

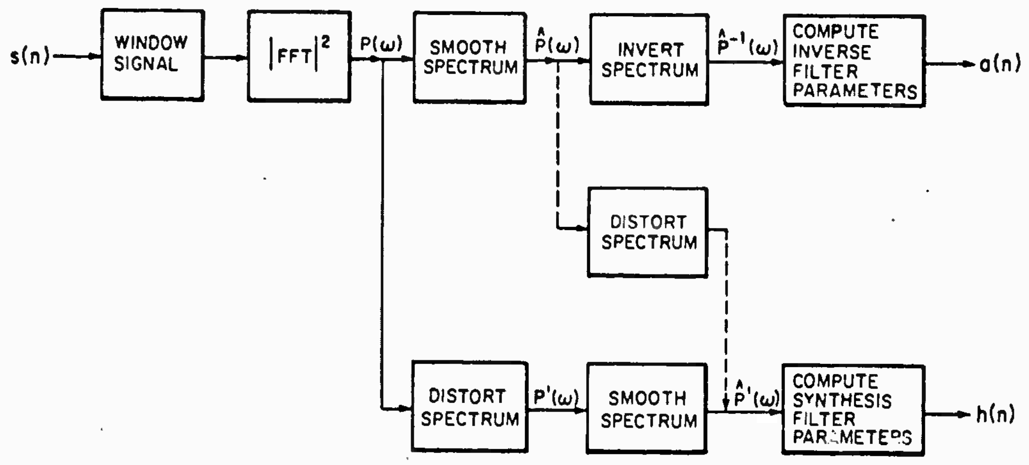

Fig. 2. Computation of the inverse and synthesis filter parameters.

The impulse response h(n) of the synthesis filter H(z) can be computed using the lower branch in Fig. 2. The signal spectrum is distorted then smoothed to obtain $\hat{P}'(\omega)$. Again, assuming a minimum phase implementation, h(n) can be computed from $\hat{P}'(\omega)$ using the cepstral method. An alternative method to compute $\hat{P}'(\omega)$ is shown by the dashed lines in Fig. 2, where the smoothed spectrum $\hat{P}(\omega)$ is directly distorted. Note that the two alternative methods do not result in identical spectra for $\hat{P}'(\omega)$. Which method to use depends on the particular application.

Since the method to compute the minimum phase impulse response from a spectrum involves taking the DFT, it is desirable for efficiency purposes to have the frequency values in the spectrum be equally spaced and their number be an integral power of 2, so that one can make use of the FFT. If $P(\omega)$ is computed using the FFT, then the two conditions can be easily met for $\hat{P}^{-1}(\omega)$. However, because of the spectral distortion in the lower branch of Fig. 2, the spectral values of $\hat{P}'(\omega)$ will not be equally spaced in general. By simple interpolation in the frequency domain, the spectral values can be computed at equally spaced frequencies, thus opening the way to the use of the FFT.

The smoothing of the signal spectrum to obtain the spectral envelope can be done in many different ways. Here, we give two popular nonparametric methods which comprise two of the three methods of spectral warping that are presented in the paper. A parametric method is given in the next section.

(a) Autocorrelation Warping

In this method the spectrum is smoothed by applying a window to the autocorrelation. This is the well-known method of spectral estimation used by statisticians [6].

(b) Cepstral Warping

In this method the log spectrum is smoothed by applying a window to the cepstrum. This method of spectral smoothing has been used extensively in speech analysis [5].

4. Linear Predictive Warping

We have called the types of spectral warping in the previous section "nonparametric" because no specific model is used to determine the impulse response of the inverse and synthesis filters. In this section we use the all-pole linear prediction model as a basis to determine the parameters of the two filters.

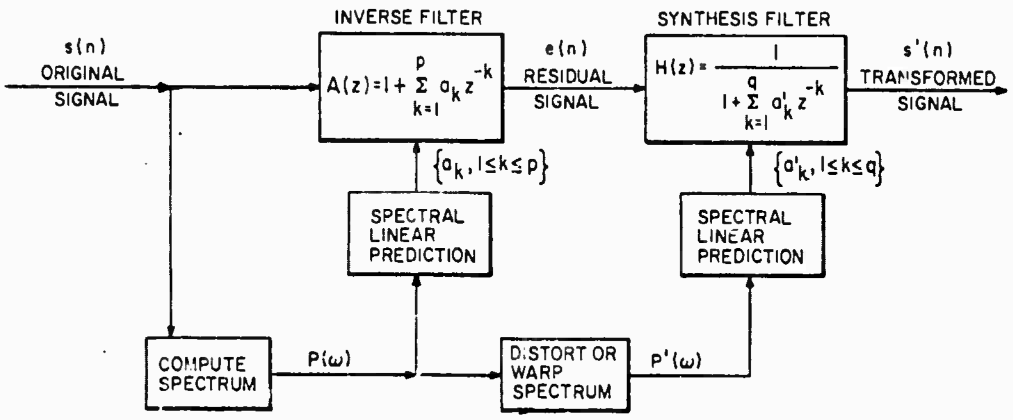

Fig. 3. Analysis-synthesis system for linear predictive warping.

Fig. 3 shows a schematic diagram of linear predictive (LP) warping. The parameters a(k) of the inverse filter are simply the predictor coefficients which are obtained by spectral LP [7] as a solution to the set of linear equations:

$$\sum_{k=1}^{p} a(k)\, R(i-k) = -\, R(i),\ 1\le i\le p, \qquad (1)$$

where p is the number of poles in the model, and R(i) is the autocorrelation of the signal, which can be computed either by taking the FFT of the spectrum $P(\omega)$, or directly from the signal. Note that the method of spectral LP inherently smoothes the signal spectrum, with the degree of smoothing being controlled by the number of poles p. Referring to Fig. 2, the smoothed spectrum $\hat{P}(\omega)$ in this case is given by the all-pole model spectrum:

$$\hat{P}(\omega) = \frac{1}{\left|1+\sum_{k=1}^{p} a(k)e^{-jk\omega}\right|^2} . \qquad (2)$$

The parameters $a'(k)$ of the synthesis filter are obtained as a solution to a set of equations analogous to (1) with $a(k)$, $R(i)$ and p replaced by $a'(k)$, $R'(i)$ and q, respectively, where $R'(i)$ is the Fourier transform of the distorted spectrum $P'(\omega)$, and q is the number of poles in the synthesis filter. In general, $q \neq p$, and its choice depends on the application.

The parameters $a(k)$ need not be computed using spectral LP, which is essentially equivalent to the autocorrelation method of LP. Instead, one could use the covariance, lattice or covariance lattice methods [8]. In that case, $P(\omega)$ is undefined. So following the dashed line in Fig. 2, we compute $\hat{P}(\omega)$ from (2), distort it, then apply spectral LP to the resulting distorted spectrum in order to evaluate the coefficients $a'(k)$ of the synthesis filter.

5. Applications

There are many possible applications for the methods of nonlinear spectral warping given above. Below, we shall give four applications: two of these use the spectral warping for a more efficient representation of the spectrum, and two are analysis-synthesis systems for generating speech that is spectrally distorted.

(a) Efficiency of Spectral Representation

In applications such as speech recognition and speech compression, it is more important to represent the spectrum accurately at low frequencies (<3 kHz) than at high frequencies (>3 kHz). Normally, anywhere between 17-20 poles are needed for an all-pole LP representation of speech spectra with a bandwidth of 7.5 kHz (sampling frequency 15 kHz). Using LP warping, for example, with frequencies above 3 kHz being heavily warped, one could have a good representation using only 12-14 poles. In this manner, one could still perform accurate formant extraction for the first three formants, with the higher formants being represented by wide spectral peaks, which is all that is usually needed [9].

For speech compression, this enables one to have wide-band, high quality speech at low bit rates, since fewer coefficients need to be transmitted. This idea has been recently implemented in an LPCW vocoder: an LPC vocoder with spectral warping [10].

(b) Unscrambling of Helium Speech

In order to render speech spoken in a helium-oxygen mixture more intelligible, it is necessary to compress the bandwidth from about 12 kHz down to 5 kHz. In addition to this linear warping, one might need to perform additional nonlinear warping at low frequencies to compensate for high pressure effects [1,2,4]. Heretofore, such nonlinear warping had not been possible.

Since the bandwidth is reduced to 5 kHz, one must still define values for the spectrum between 5 and 12 kHz (assuming a 24 kHz sampling frequency). The reason is that in our analysis-synthesis system the sampling rate remain fixed. It is usually sufficient to assign a positive constant for the spectrum between 5 and 12 kHz that is a fixed number of decibels below the maximum value in the spectrum. A value of zero, however, is not recommended.

(c) Speech for the Hard of Hearing

Many people with severe hearing loss cannot hear frequencies much above 1 kHz [11]. An idea that some researchers have had is to compress the speech spectrum so that the most important part of the spectrum (up to 3 kHz) is compressed down to less than 1 kHz. It is hoped that this squeeze of the spectral information into a small bandwidth would aid the hard of hearing in listening to speech, and would eventually lead to the design of more effective hearing aids. It is easy to show that a simple linear compression of the spectrum to less than 1 kHz is quite unintelligible. However, the results improve dramatically if a nonlinear warping that emphasizes low frequencies is effected.

The technical details for this application are very similar to those described above for the unscrambling of helium speech.

6. Conclusion

A general analysis-synthesis system for the nonlinear spectral distortion of speech signals was described. The method does not need any pitch extraction, and allows for the arbitrary specification of the warping function. The latter is accomplished by performing the warping directly in the frequency domain. Depending on the type of spectral smoothing used, three methods resulted: autocorrelation, cepstral and linear predictive warping. Applications for these methods included bit rate reduction in high quality speech compression systems, efficient spectral representation for use in speech recognition systems, unscrambling of helium speech, and spectral compression for the hard of hearing.

References

[1] F. Quick (1970)
"Helium Speech Translation Using Homomorphic Techniques," S.M. Thesis, M.I.T., Cambridge, Mass., June.

[2] V. Zue (1971)
"Translation of Divers' Speech Using Digital Frequency Warping," QPR No. 101, R.L.E., M.I.T., Cambridge, Mass., 175-182, April.

[3] A. Oppenheim and D. Johnson (1972)
"Discrete Representation of Signals," Proc. IEEE, Vol. 60, 681-691, June.

[4] H. Suzuki, G. Ooyama and K. Kido (1974)
"Analysis-Conversion-Synthesis System for Improving Naturalness and Intelligibility of Speech at High-Pressure Helium Gas Mixture," Preprints, Speech Communication Seminar, Stockholm, Vol. 1, 97-105, August.

[5] A. Oppenheim and R. Schafer (1975)
<u>Digital Signal Processing</u>, Ch. 10, New Jersey: Prentice-Hall.

[6] R. Blackman and J. Tukey (1958)
<u>The Measurement of Power Spectra</u>, New York: Dover.

[7] J. Makhoul (1975)
"Spectral Linear Prediction: Properties and Applications," IEEE Trans. Acoustics, Speech and Signal Processing, Vol. ASSP-23, 283-296, June.

[8] J. Makhoul (1976)
"New Lattice Methods for Linear Prediction," IEEE Int. Conf. Acoustics, Speech and Signal Processing, Philadelphia, April.

[9] S. Itahashi and S. Yokoyama (1974)
"A Method of Formant Extraction Utilizing the Mel Scale," J. Acoust. Soc. Japan, Vol. 30, No. 12, 677-678, December.

[10] J. Makhoul and L. Cosell (1976)
"LPCW: An LPC Vocoder with Linear Predictive Spectral Warping," IEEE Int. Conf. Acoustics, Speech and Signal Processing, Philadelphia, April.

[11] J.D. Schein and M.T. Delk, Jr. (1974)
<u>The Deaf Population of the United States</u>, National Association of the Deaf, Silver Spring, Maryland.

C. Acoustic-Phonetic Recognition in BBN SPEECHLIS

Richard M. Schwartz
Victor W. Zue

This paper describes the acoustic-phonetic analysis of continuous speech in a complete speech understanding system. The system accepts various parameters derived from the digital waveform and short-time spectra, and produces a segment lattice where segments can have overlapping boundaries and the description of segments is a list of labels. Acoustic-Phonetic as well as phonological knowledge of English is employed extensively in labeling the segments. Each label also has associated with it a score, reflecting the confidence in its identity. A description of the acoustic-phonetic analyzer, as well as statistics related to its performance will be presented in detail.

1. Introduction

One phase in an Automatic Speech Understanding System is Acoustic-Phonetic Recognition (APR), which uses time varying acoustic parameters (e.g., energy, formant frequencies) to make a best attempt to transcribe an utterance. The results are then used by a lexical retrieval component (word matcher) [Klovstad, 1975] to determine which words (in the lexicon) might be in the utterance.

The first section of this paper puts forth the basic philosophies used in the APR component of the BBN Speech Understanding System (SPEECHLIS). The second section describes the state of this component as of December 1975. The third section reports some performance statistics for the APR component as of the same date.

2. Basic Philosophy

(a) Segment Lattice

Acoustic-Phonetic Recognition in SPEECHLIS consists of two basic tasks: SEGMENTATION and LABELING. SEGMENTATION is deciding where the phoneme boundaries are, and LABELING is deciding the phonetic identity of the segments produced by the segmentation phase. It should be noted that the distinction between the two phases is not always clear cut.

The most important aspect of the APR philosophy is the use of a SEGMENT LATTICE. The use of a SEGMENT LATTICE reduces the chance of segmentation errors by providing alternate segmentation paths. For further definition and justification of the use of the SEGMENT LATTICE see Schwartz [1975].

(b) Multiple Pass Strategy

Because the acoustic characteristics of a phoneme vary greatly with its context, it is very helpful to be aware of the nature of that context when making any decision in the construction of a SEGMENT LATTICE. One way to make use of contextual information is to employ a Multi-Pass APR strategy. Each pass consists of four steps: initial segmentation, initial labeling, adjustment of boundaries, and relabeling. Boundaries are adjusted so that they correspond to reliable acoustic events. The acoustic events examined are determined by the results of the initial labeling. Relabeling is then performed using the adjusted boundary times. Each pass operates on regions generated by the segmentation in the previous pass, performing more detailed segmentation and labeling that use more detailed contextual information. In this way, acoustic-phonetic rules can be designed for specific phonetic environments.

(c) Reliable Boundary Confidences

While adding a few optional paths to a SEGMENT LATTICE greatly increases the probability that the "correct" path is represented, it also increases the ambiguity facing the word matcher. In order to partially alleviate this problem, it is helpful to include a confidence measure for each boundary in the lattice. If the boundary confidences are combined with the scores assigned to word matches, and if the boundary confidence scores are well-correlated with the likelihood that the boundary is part of the "correct" path, then the word matcher will be better able to choose between the many alternate paths in the SEGMENT LATTICE. In order to compute a confidence on each boundary, a parameter relevant to the evidence of a boundary should be used. For instance, the depth of a dip is a good indicator of the reliability of that dip as a boundary.

(d) Experimentation

The parameters and threshold used by the SEGMENTATION and LABELING program are determined from actual data in a data base with the aid of an experiment facility described elsewhere in these proceedings [Schwartz, 1976]. This approach assures that the algorithms developed are realistic and within the capabilities of a computer program.

(e) Probabilistic Labeling

An important issue in the design of an APR program is how it will interface with a word matching component of a total speech understanding system. Though true probabilities can be hard to estimate accurately, they afford a well-defined formalism for manipulation and combination of scores. Consequently - in an effort to provide the word matcher with the maximum amount of relevant information about each segment - a labeling philosophy to characterize directly each segment probabilistically has been adopted. This is contrasted with the philosophy of explicitly labeling each segment as a single allophone or phoneme.

These two philosophies differ in a way which may not be immediately evident. In the first case probability distributions (one for each allophone) which depend on the values of the observed acoustic parameters are evaluated to produce scores for the different allophones. The specific values of the parameters observed in each segment are used in these evaluations. The segment characterization produced by the APR (and presented to the word matcher) for each segment in the SEGMENT LATTICE is a vector of computed scores (probabilities) with one element per allophone.

In the second case the APR provides only a single label, which may be interpreted to represent a single phoneme or a larger class of phonemes. This can be thought of as its observed acoustic characterization. In this case, however, an interface between the APR and the word matcher effectively provides the desired scores by consulting a confusion matrix which contains probabilities for every combination of allophone and segment label. As long as variations in the relevant acoustic parameters do not cause a segment label change, none of the scores provided to the word matcher by the interface will change. However, this is contrary to the observation that variations of acoustic parameters for a single phoneme do in fact change the confusion likelihood of that phoneme with other phonemes.

The first philosophy results in a better characterization of the segment because relevant parameter variations otherwise ignored (e.g. whenever the parameter variations would not have caused a segment label change) can be incorporated in the word matcher scoring mechanism. Since this technique requires evaluating all possibilities, it is more costly, however. Therefore, what we have chosen is a flexible combination of the two techniques. For those phonemes which are very unlikely to match a particular segment, the probabilities predicted by a long term confusion matrix are a good approximation to the likelihoods which would be computed explicitly. For example, if one believes a segment to be a [t], the probability distributions for [t,p,k,d,n] should be evaluated using the observed parameters. But the scores on each of the vowels are all bad, so they will be fairly insensitive to this particular manifestation of [t]. This means that not all scores in the vector need be computed for every phoneme label on each segment; most can come from the confusion matrix, while those that are sensitive to parameter variations will be computed individually. In this way, we can start with only a few of the scores in a probability vector being modified, and gradually increase the number of modifications as more specific results are obtained from our experiment facility [Schwartz, 1976].

(f) Context Dependency

In using context when labeling a segment, it would be very helpful to know, with absolute certainty, the identity of the adjacent segments. However, if context is used, then incorrect hypotheses about the identity of the adjacent segments could lead to labeling errors. In those cases where these hypotheses are more likely to be incorrect, it would be advantageous to consider all possible relevant contexts, and compute different results for each postulated context. For example, one way to distinguish between the unvoiced plosives [p,t,k] is to examine the burst frequency and the voice-onset-time (VOT) when the plosive is followed by a vowel or semi-vowel. However, if the unvoiced plosive is followed by [R], then the burst frequency and VOT change considerably. Since some of the [R] transition is often unvoiced when it follows an unvoiced plosive, it is not always possible to determine (absolutely) whether the plosive is followed by a vowel or by [R]. Therefore, we must consider two (or more) allophones of each plosive; one followed by vowels, the other followed by [R]. Then the score on [T-R], for example, is the probability that the relevant acoustic parameters (voice onset time, burst spectrum, formant motions, etc.) would have the values they do, given that the phoneme this segment represents is a [T] and given that it is followed by [R]. When used in word matching, only the score of the appropriate allophone of [T] need be examined, determined by the actual spelling of the word being considered. Of course, one wants to minimize the number of different allophones that need to be considered, but a reasonable balance can result in a large improvement in word matching.

3. Acoustic-Phonetic Recognition Program

A program which embodies most of the philosophies described in the first part of this paper has been developed. The program starts by looking at dips in three energy parameters to form a preliminary SEGMENT LATTICE. Using a general dip detection routine, dips are found in three wide band energy parameters (tabulated below). Sonorant regions are separated from obstruent regions using the dips in the low frequencies. Then, dips found in the middle and high frequencies are taken as an indication of possible nasals, glides, or voiced obstruents (e.g. V,DH,HH,DX). Within regions initially classified as obstruent, the dips in the high frequency band are used to separate strident fricatives from plosives and weak fricatives.

This preliminary segment lattice has little or no branching and only distinguishes among a few broad categories. Some of the regions generated by this initial phase contain more than one phoneme, but within each region, parameters specific to the type of region are used for further segmentation and labeling.

Next, a sequence of 26 ordered Acoustic-Phonetic rules is applied to the lattice. Each rule only applies at certain places in the lattice, which depend on the partial segmentation and labeling. Using the labeling in the lattice at that time and the acoustic parameters, the rules can delete branches, expand the lattice by adding branches, and change or narrow the label on any segment. For example, one rule only applies to sequences of two segments initially labeled "obstruent-fricative". If the minimum energy during the obstruent is low enough, and the fricative is short and weak, then the sequence is bridged with an unvoiced plosive. Depending on the actual parameter values, the original path may be deleted. It is of some interest that the ordering of the rules, so far, has been relatively straightforward.

The program currently attempts to distinguish between all vowels and diphthongs, unvoiced plosives, intervocalic glides and liquids, strident and weak fricatives, affricates, and flapped dentals. It also detects and labels prevocalic and postvocalic glides, sentence initial [HH] or glottal stops, and unreleased plosives at the end of a sentence. Formant transitions are used to separate postvocalic velar consonants from non-velars, and in some cases, labials from non-labials. The program also detects unreleased plosive-plosive pairs, pauses, syllabic nasals, vowel-schwa pairs (such as IY-AX in "give me a list"). The program associates with each segment a number which is equal to the maximum energy in the segment. This is used by the word matcher to evaluate within-word stress differences. The APR component currently requires approximately three times real time to generate a segment lattice from the acoustic parameters for an utterance, including the time necessary to read in all the parameters.

The parameters currently used by the program follow. The "Z" at the end of three parameter names indicates that the parameter has been smoothed by a 3-point (1/4-1/2-1/4) zero-phase filter.

Name	Definition	Use of Parameter
LEZ	Smoothed energy in the region from 120-440 Hz.	Sonorant obstruent segmentation, Aid in voicing decision on fricatives
MEPZ	Smoothed energy in the preemphasized spectrum from 640-2800 Hz.	Segmentation of non-vowels within sonorant regions.
HEPZ	Smoothed energy in the preemphasized spectrum from 3400-5000 Hz.	Segmentation of non-vowels within sonorant regions. Unvoiced-Plosive detection.

ROP	Energy in the preemphasized spectrum.	Burst location, Plosive and Fricative identification, and many others.
F1 F2 F3	Formant Frequencies	Detecting glides and nasals within sonorant regions, segmenting vowel regions, labeling vowels and glides, some consonants.
F0	Fundamental Frequency	Normalizing formants, aid in voicing decision.
CM75	"75% center of mass" indicates rough spectral shape.	Used in identifying fricatives and unvoiced plosives.

Figure 1 shows a plot of a SEGMENT LATTICE with energy and formants plotted below. Segment labels are shown at the beginning of the corresponding boundary marker. It is important to note that though some of the segment labels used appear to be from the same set (ARPABET) as our dictionary symbols, they, in fact, represent a vector of a long term confusion matrix. The sentence analyzed is "List all trips already taken."

4. Program Performance

Since we are dealing with a lattice of segments, each labeled with a vector of probabilities of phonemes, we must devise several ways to measure performance in terms of accuracy and specificity.

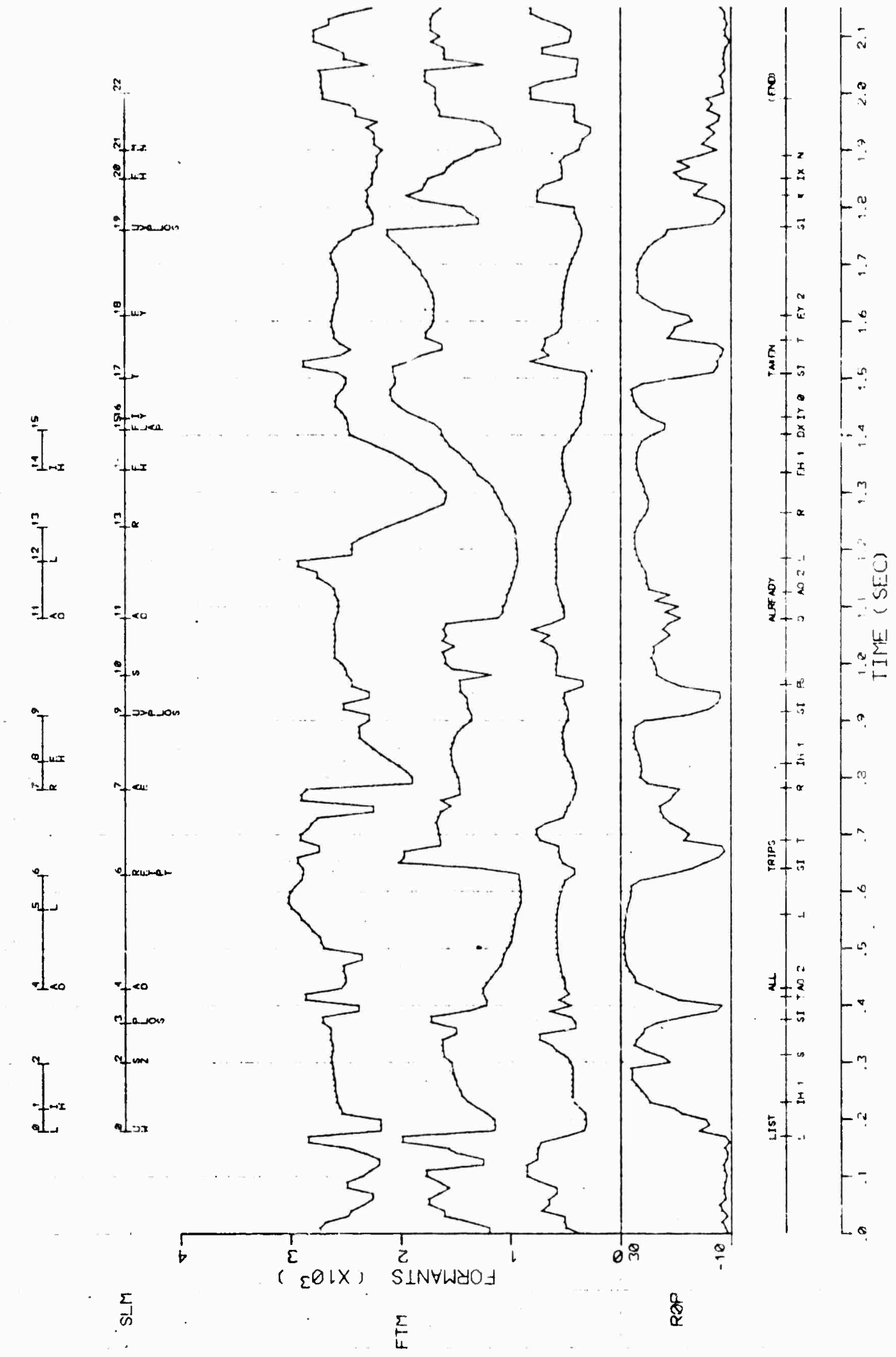
SLM
FTM
RØP
FORMANTS (X10³)
TIME (SEC)
LIST
ALL
TRIPS
ALREADY
TAKEN
(END)

Fig. 1

The following definitions of the SEGMENT LATTICE will be used.

Best Path: The sequence of contiguous segments through the utterance which results in the smallest number of segmentation errors.

Number of missing segments: The number of phoneme boundaries in the ideal segmentation which were not found in the acoustics at all. In other words, two phonetic segments being merged into one.

Number of extra segments: The number of extra segments in the best path of the SEGMENT LATTICE. In other words, calling one segment two or more phonetic segments.

Branching Ratio: The number of segments divided by the number of boundaries, i.e., the average "depth" of the lattice.

The data shown below is gathered from 38 sentences spoken by three male speakers.

Total number of segments in ideal segmentation	1125
Total number of boundaries in lattices	1324
Total number of segments in lattices	1680
Average Branching Ratio	1.27
Total number of missing segments	31
Percentage of missing segments (approximately one for each sentence)	2.4%
Total number of extra segments	39
Percentage of extra segments (approximately one for each sentence)	3.1%

Since the format of a segment label consists of a score (likelihood ratio) for each of the dictionary symbols possible (currently 60), it is not appropriate to measure "labeling errors". However, we can measure selectivity of the labels in a few different ways. For instance, one good measure is the percentage of the time that the correct phoneme has the highest score of all the phonemes, or, in general, the likelihood that the correct phoneme is within the top n choices. This measure of labeling performance is highly dependent on the number of allophones defined in the system. For a system with fewer than 60 allophones defined, one would expect that the correct choices would be closer to the top. The following table shows the percentage of the time that the correct phoneme is within the top n choices, where n is the column. For example, 90% of all the segments were labeled correctly within the top 5 choices. The first row is for all phonemes, and the second is just for vowels and diphthongs. In comparing these results with those cited in the literature, it should be noted that the number of possible choices are quite large. For example, the program attempts to distinguish between 26 different vowels and diphthongs.

Choices	1	2	3	4	5	6	7	>7
All phonemes	56	72	81	87	90	91	94	100
Only vowels	48	66	81	86	89	90	92	100

4. Conclusion

We have described the acoustic-phonetic recognition component of the current BBN Speech Understanding System (SPEECHLIS). Although this component is still under active development, it has been incorporated into SPEECHLIS. Preliminary results indicate that the program performs well under field conditions.

References

[1] J. Klovstad and L. Mondshein (1975)
"The CASPERS Linguistic Analysis System," IEEE Trans. on Acoust., Speech, and Signal Processing, Vol. ASSP-23, No. 1, February.

[2] R. Schwartz and J. Makhoul (1975)
"Where the Phonemes are: Dealing with Ambiguity in Acoustic-Phonetic Recognition," IEEE Trans. on Acoust., Speech, and Signal Processing, Vol. ASSP-23, No. 1, February.

[3] R. Schwartz (1976)
"Acoustic-Phonetic Experiment Facility for the Study of Continuous Speech," elsewhere in these proceedings.

D. Acoustic-Phonetic Experiment Facility For The Study of Continuous Speech

Richard M. Schwartz

While gathering acoustic data for the acoustic-phonetic analysis of speech, it is necessary to consider many different sounds in varying phonetic environments to assure that the results are statistically significant. In order to reduce the amount of time required to test hypotheses, a facility has been developed which provides an interactive environment for performing a wide variety of acoustic-phonetic experiments on a large data base of continuous speech. Using this facility, one can formulate an experiment, run it on selected portions (or all) of the data base, and display or tabulate the results in a meaningful way. Another experiment may then be run based on the results. CPU time required to run an experiment on the entire data base is between 5 and 20 seconds, depending on the complexity of the experiment. Due to the ease of interactions, formulating or revising an experiment, running it, and displaying the results normally takes less than 5 minutes. This facility has been used in combination with a data base of 69 hand-labeled sentences to develop algorithms for acoustic-phonetic segmentation and labeling in a speech understanding system. Several examples of its use and the results obtained will be given.

1. Introduction

Doing an acoustic-phonetic experiment by hand on more than a small data base is a tedious, time consuming and error prone process. In order to leave the speech researcher free to examine results and to make conclusions or try other experiments, we have developed a highly interactive computer facility which enables us to specify a phonetic context, introduce a general algorithm or procedure, specify which of the utterances in the data base are to be scanned (if not all of them) and present the results in a meaningful way. Due to ease of interaction, this entire experiment process can usually be performed in less than five minutes. It is then possible to modify the algorithm, or change the phonetic environment, and rerun the experiment on the same or different data, and display the new results in a shorter amount of time.

This makes it possible for us to develop many different algorithms in order to incorporate as much acoustic-phonetic knowledge as possible in our speech understanding system.

Our data base currently consists of approximately 100 sentence length utterances of continuous speech. There is, for each utterance, a careful manual transcription consisting of time markers, phonetic symbols, stress marks, word and syllable boundaries and orthographic spellings; and also, a set of approximately 40 time varying acoustic parameters such as energy, formants, bandwidths, zero-crossings, etc. with values computed once for each 10 msec frame.

Two examples of the types of experiments which can be done follow.

2. Intervocalic W and L

First, we will consider the problem of distinguishing between [W] and [L] in intervocalic position. Though the first two formants of [L] are often higher than those of intervocalic [W], the two formant values are not sufficient to completely characterize the two. However, there are other differences. Among these differences are: the third formant of [L] is usually much higher than that of [W], and the first formant of [L] often exhibits a sharp discontinuity between the [L] and the following vowel.

In order to perform an experiment, we must first specify the context of interest. The context is specified as a sequence of SEGMENTS, each defined by a boolean combination of features, phones, stress levels, and orthographic spellings, allowing specification of optional segments and word or syllable boundaries. The context shown in Fig. 1 searches for any vowel, followed by [W] or [L], followed by any other vowel. An algorithm or list of functions which is to be computed for every occurrence of this phonetic context is then entered.

There are 34 different function types to choose from, including arithmetic and boolean functions, conditional branching operations, parameter manipulation routines, and several others - in short, enough to allow implementation of a wide range of algorithms. The experiment shown only uses three of the simpler parameter manipulation routines. These functions are entered using a relatively restricted command language designed to suggest English phrases or sentences. The program interprets them as they are typed in, checking for consistency between arguments and prompting for successive arguments. The arguments to each of the functions may be the result of lower numbered functions, and those functions which are associated with a text name may be referred to by that name.

The first function, named "Target," searches through the second formant track in the region labeled as the second segment (the [L] or [W]) in the context. It returns as its value, the time that the formant was at its minimum.

The experiment is then run for any subset of the data base, which is cross referenced in several ways (e.g., by speaker, sentence number, sex, date of recording, etc.). The actual scan and computation for the entire data base requires about 15 seconds. Once the data has been gathered and stored internally, the user has several options. One option is to enter another interactive phase of the facility which allows examination of the results in several ways. The scatter diagram in Fig. 2 shows the target first formant (F1) and maximum third formant (Max F3) values for each sample. All figures in this paper are taken from actual displays produced by the program.

Since none of the occurrences of [W] have a high third formant, the user might eliminate those [L]´s above 2500 Hz from the picture, in order to get a closer look at the remaining samples. It is possible to edit, delete, insert or add new functions to the existing algorithm. In this case, one function is added to reject any sample with a maximum third formant greater than 2500 Hz, as if the context did not match. The value of F1 and the maximum first difference of F1 (MAX DF1) is shown for the remaining samples in Fig. 3. Most of the occurrences of [W] have a low F1 and low Maximum DF1. This leaves just 7 of the original 39 samples. Using this facility, one can then search for additional features which further separate the two phonemes. Figure 4 is a three dimensional scatter diagram using the same three acoustic features, rotated to show the maximum separation in two dimensions. All but two of the [L]´s are clearly in the cluster on the right.

3. Retroflexed Unvoiced Plosives

Figure 5 shows an experiment entered into the facility in order to examine some of the acoustic correlates of unvoiced plosives in varying contexts. Although the burst times are indicated in the hand labeling, this algorithm attempts to locate the burst, using the energy in the preemphasized signal (ROP). Once the burst is found, the voice onset time (VOT) and burst frequency (in terms of the parameter named CM75) are measured. To determine the effect of retroflexion on unvoiced plosives, one can easily compare the burst frequency and voice onset time of prevocalic unvoiced plosives (Fig. 6) with those same parameters for unvoiced plosives followed by [R] (Fig. 7). To emphasize the effect on [T], Fig. 8 compares the burst frequency and VOT for retroflexed [T]´s and the prevocalic [T]´s. The "R"s represent the retroflexed [T]´s and the "V"s represent the prevocalic [T]´s. Though this data is gathered from sentences of 4 male speakers, there is a clear separation between prevocalic and retroflexed [T]´s.

Since the scoring and decision strategies in the BBN Acoustic-Phonetic Recognition Program and Lexical Retrieval Component are based on the Bayesian probability of a sequence of dictionary symbols in light of the acoustic evidence, we would like to be able to estimate probability distributions. Figs. 9a-9d show one dimensional density distributions of the burst frequency for [T] and [K] followed by vowels or [R]. We have several smoothing functions, some of which are parametric [1]. Beta distributions are very useful for fitting one dimensional, asymmetric density distributions. These are shown superimposed on the histograms. Fig. 10 shows the estimated probability density function (and the Beta distribution fit) for the burst frequency of all unvoiced plosives followed by vowels or [R]. Using the five parametric models, (Figs. 9 & 10) we can determine the likelihood ratios for each of the 4 cases with respect to the acoustic feature - burst frequency. For example, for the case of [T] followed by [R] the likelihood ratio computed from the data in figures 9b and 10 is:

$$\frac{P(\text{burst frequency} \mid [\text{T-R}])}{P(\text{burst frequency} \mid \text{unvoiced plosive})}$$

In general, (as in this case) there several dependent dimensions. We are currently working on methods for modeling multi-dimensional, asymmetric (non-Gaussian) distributions.

The program allows 12 different types of graphical displays, including 1-dimensional or rotatable 2-dimensional histograms, density distributions, or cumulative distributicns; and 2-dimensional or rotatable 3-dimensional scatter diagrams. During an experiment session, any combination of the previous 15 displays can be superimposed for easy comparison (e.g., Figs. 9 & 10). The program also allows 13 different kinds of listings and tabulations of results.

The two experiments which were demonstrated illustrate the use of this facility for well-known acoustic phenomena. In general, one can easily perform more complex experiments for determining the acoustic correlates of specific phonemes in a variety of phonetic contexts. We feel the use of this system facilitates the design of phonetic segmentation and labeling algorithms to be used in the analysis component of our speech understanding system [2].

References

[1] J. Makhoul and R. Schwartz (1975) "Parametric Modeling of Probability Distributions," in Speech Understanding Systems, Annual Technical Report, BBN Report No. 3188, pp. 50-65, November.

[2] R. Schwartz and V. Zue (1976) "Acoustic-Phonetic Recognition in BBN SPEECHLIS," elsewhere in these proceedings.

```
Try to separate intervocalic L and W
For
Context: (vowel) (W L) (vowel)
  Tabulate:
 1) Target:
   Time of minimum of parameter F2M from start to end of segment #: 2
 2) F1:
   Value of parameter F1M at Target
 3) F2:
   Value of parameter F2M at Target
 4) F3:
   Value of parameter F3M at Target
 5) Max DF1:
   Maximum of parameter Derivative of F1M from
         first quarter of segment #: 2 until center of segment #: 3
 6) Max F3:
   Maximum of parameter F3M from start to end of segment #: 2
 7) Reject this occurrence if: Max F3 is greater than 2500
```

Fig. 1. Intervocalic L & W

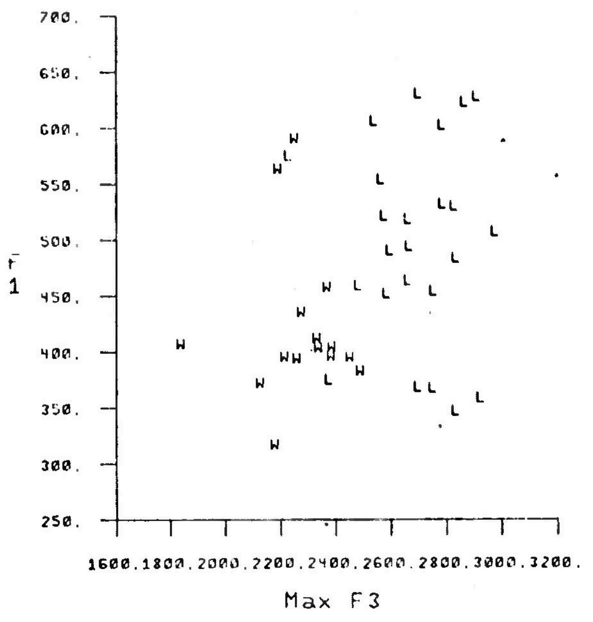

Fig. 2. Max F3 vs F1 for W and L

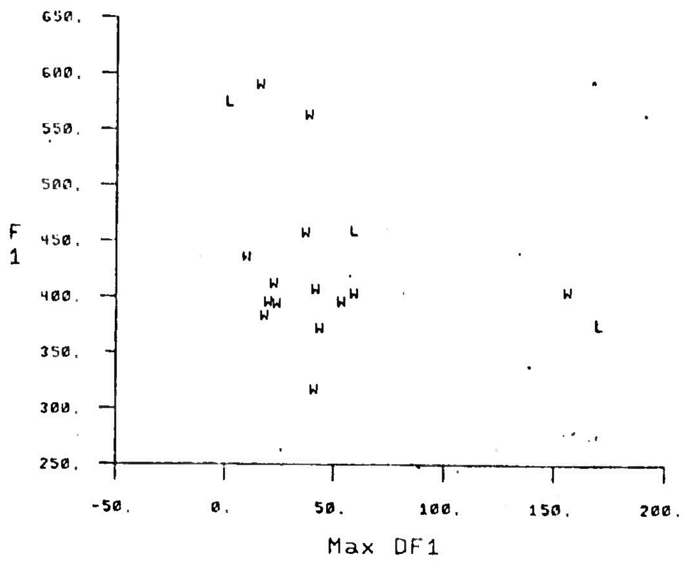

Fig. 3. F1 vs Max DF1 for W and L

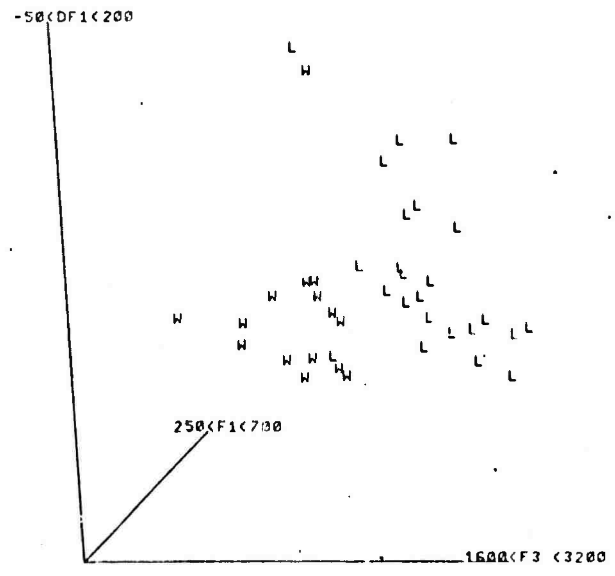

Fig. 4. 3-D Max F3 vs Max DF1 vs F1

```
Find Burst, Measure CM75,VOT for PTK
For
Context: (SI) (unvoiced.plosive) (vowel R)
  Tabulate:
 1) T.Min ROP:
   Time of minimum of parameter ROP from
        start to end of segment #: 1
 2) Difference of start of segment #: 2 and
        T.Min ROP
 3) Max DROP:
   Time of maximum of parameter Derivative of ROP
        from T.Min ROP until center of segment #: 2
 4) Difference of start of segment #: 2 and Max DROP
 5) MinROP:
   Value of parameter ROP at T.Min ROP
 6) ROP at Max DROP:
   Value of parameter ROP at Max DROP
 7) DROP:
   Difference of ROP at Max DROP and MinROP
 8) Threshold:
   Sum of MinROP and 7
 9) Next time that parameter ROP from T.Min ROP until
        first quarter of segment #: 3 is greater than
        Threshold
 10) Boolean Result of: value #9 is less than 0
        implies Max DROP or value #9
 11) Next time that parameter 2nd Derivative of ROP
        from value #10 until first quarter of
        segment #: 3 is less than 0
 12) Burst:
   Difference of value #11 and 1
 13) VOT:
   Difference of end of segment #: 2 and Burst
 14) CM75 at Burst:
   Value of parameter CM75 at Burst
 15) ROP at Burst:
   Value of parameter ROP at Burst
```

Fig. 5. Prevocalic Unvoiced Plosives

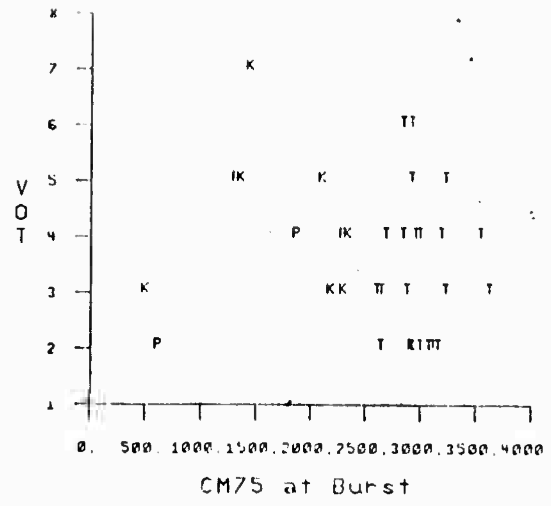

Fig. 6. CM75 vs VOT for PTK-vowel

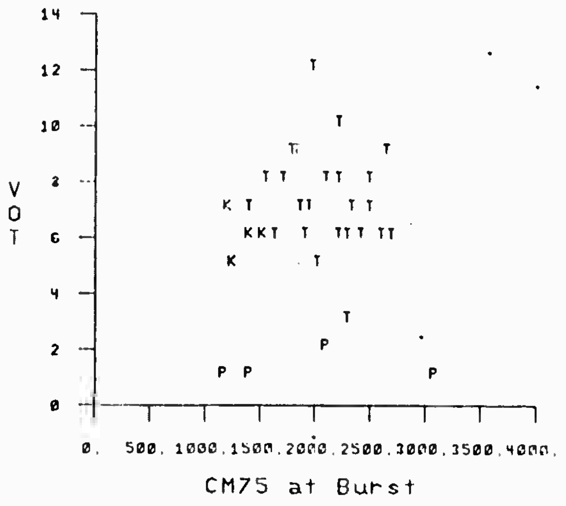

Fig. 7. CM75 vs VOT for PTK-R

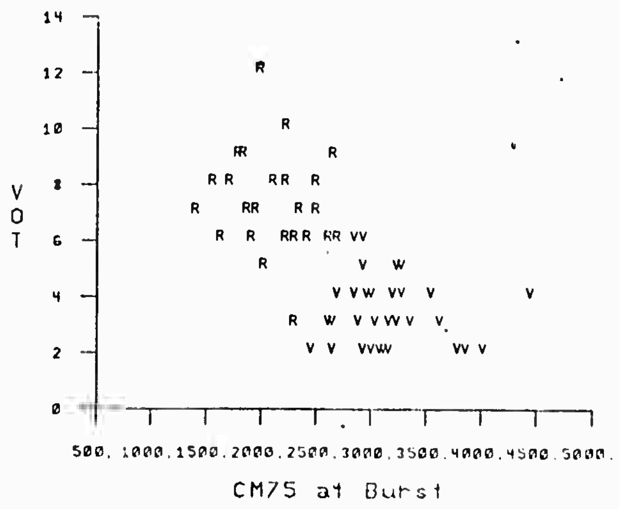

Fig. 8. CM75 vs VOT for T-V and T-R

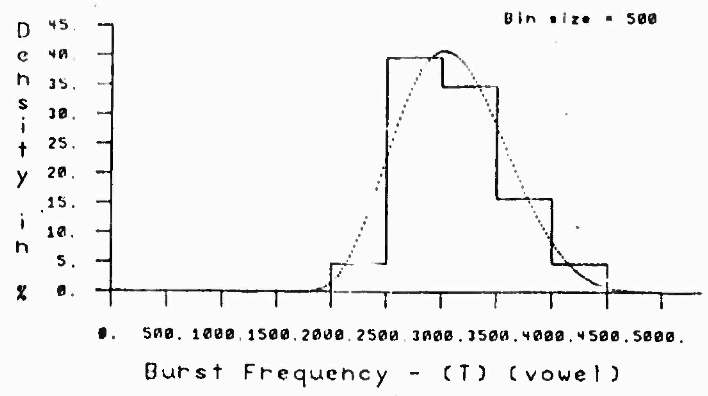

Fig. 9a. CM75 for T-V

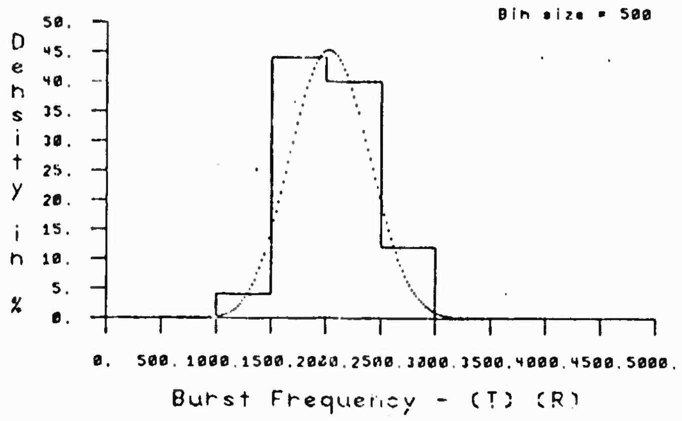

Fig. 9b. CM75 for T-R

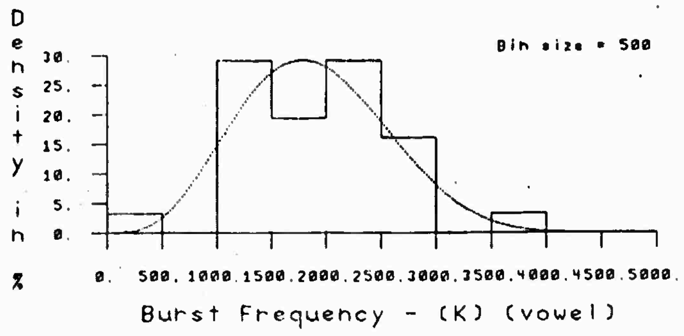

Fig. 9c. CM75 for K-V

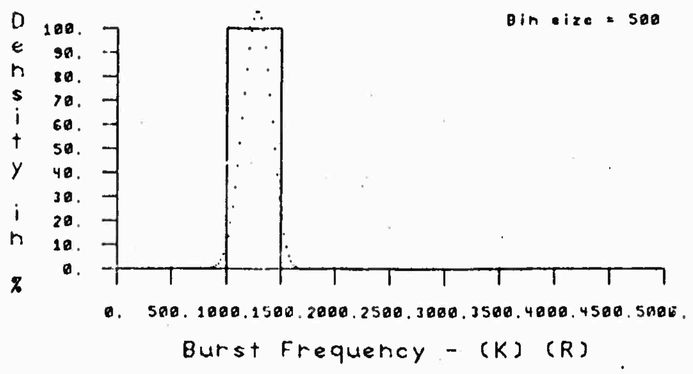

Fig. 9d. CM75 for K-R

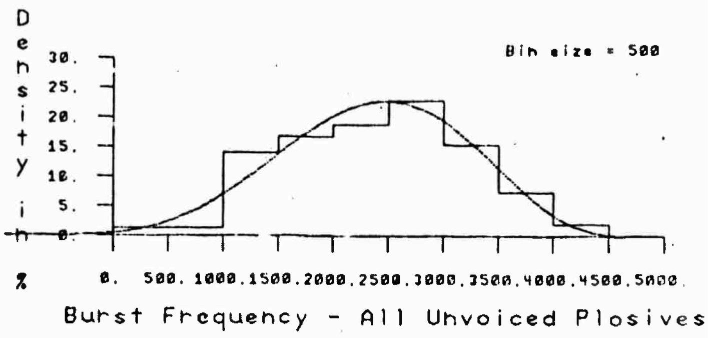

Fig. 10. CM75 for PTK-V and PTK-R

E. Word Verification in a Speech Understanding System

Craig Cook

If, in a speech understanding system, word matching is performed at the phonetic level, then the accurate determination of the locations and identities of words present in an unknown utterance is necessarily limited by the phonetic segmentation and labeling. Verification offers an alternative strategy by doing a top-down parametric word match independent of segmentation and labeling. The result is a distance measure between the reference parameterization of a hypothesized word and the computed parameterization of the real speech. This distance is interpreted as the likelihood of that word having actually occurred over a given portion of the utterance.

1. Introduction

Given the results of the Klatt-Stevens spectrogram reading experiment [1], it seems clear that the ability to return to acoustic evidence for verifying word hypotheses is important to correct recognition. This is because one can verify the consistency of all acoustic clues with respect to the given word hypothesis. Assuming that phonological and coarticulation processes are described by rules which are generative in nature, we feel that an analysis-by-synthesis procedure is needed to overcome inaccuracies present in preliminary phonetic analysis and to decode the effects of the phonological rules. The synthesis phase of a word verification component must be able to transform a broad phonetic transcription into a parametric representation suitable for comparison with the acoustic parameterization of an unknown utterance.

To this end, we have written a high-level preprocessor to translate a set of phonological and acoustic-phonetic rules into a Fortran compilable synthesis-by-rule program [2]. The synthesis program takes into account phonological effects across word boundaries, altering the parameterization according to the context in which the hypothesized word may occur. This allows us to derive in near real-time a parametric representation of any word, given its phonetic transcription.

Speech recognition systems using templates extracted from real speech have achieved impressive results. White [3] has reported on an isolated word recognition application and Bridle [4] has described a technique for word spotting. For our particular application, we have chosen to use synthetic templates generated by a synthesis-by-rule program. This choice involved several considerations which are summarized below.

a) Using templates extracted from real speech requires storing a parameterization for each entry in the lexicon, which for large vocabularies may require a substantial amount of storage. On the other hand, the generation of synthetic templates requires only the storage of the synthesis program with a relatively small number of parameters.

b) In continuous speech, it is easier to deal with contextual effects of surrounding words by using synthetic templates. This is because we can deal with them at the phonetic level (see Section 4) These effects are particularly important for short function words.

c) In a multi-speaker environment, a system using real-speech templates will perform best if it is trained on each new speaker for the whole vocabulary. This technique is not practical for systems with very large vocabularies. However, in a synthesis program, it is possible to extract the speaker dependent parameters from a relatively small speech sample.

The chief limitation of using synthetic templates is their dependence on the ability of the synthesis program to generate accurate parameterizations. Inadequacies in the program may produce incorrect results in the verification component as a whole.

The verification component includes a time normalization routine and a parametric matching program. Time normalization is done using a dynamic programming algorithm based on a method first introduced by Itakura [5]. The algorithm involves a non-linear time warping based on the registration of the error metric, in this case the log ratio of the linear prediction residuals. We have modified Itakura's method to allow limited misalignment in time between the hypothesized word parameterization and its hypothesized position in the utterance. In actually computing the distance measure between these two, we sum the error metric between corresponding segments (frames), the correspondence having already been determined by the time normalization technique.

2. Synthesis-by-Rule

To synthesize the parametric representation of a hypothesized word, we use its phonetic spelling as given by the speech understanding system's phonetic dictionary. This transcription, plus any surface structure and available semantic information, serves as the input string to the phonological phase of a synthesis-by-rule program. This phase contains all of the program's phonetic rewrite rules plus a number of acoustic-phonetic rules dealing with phonological effects. After being processed by these rules, a more detailed phonetic transcription enters the phonetic phase of the progra where a direct phonetic-to-parametric transformation is performed. The output parametric representation consists of a set of time functions which would ordinarily control a terminal analog waveform synthesizer. The synthetic waveform is not needed in this application because matching is done at the spectral level between spectra extracted fr the real speech and synthetic spectra computed from the parameters generated by the synthesis program. Parameters available for use in verification include:

a) three formant frequencies and bandwidths

b) a nasal pole-zero pair

c) amplitudes of voicing, aspiration, and frication sources

d) fundamental frequency.

The parameters currently used in word matching are a subset of those available from the synthesis program.

3. Spectral Matching

Our spectral distance measure is the log ratio of the linear prediction residuals. This metric was derived by Itakura [5] using maximum likelihood. Alternate expressions for this metric are given in Makhoul [6]. Briefly, the metric is defined as:

$$d = \log \frac{{}_{i=p}\; b'_i \; R_i}{\sum_{i=q}^{q} b_i \; R_i}$$

where {R(i)} are the autocorrelations of the matching signal and {b(i)} and {b´(i)} are the autocorrelations of the predictor coefficients for the matching and the reference signals respectively. "p" and "q" are the orders of the respective predictors. Here, the reference signal represents the real speech and the matching "signal" is represented by the synthetic spectrum. To compute the measured spectrum from the real speech, we divide the speech signal (sampled at 10 or 20 KHz) into 20 msec windows overlapped every 10 msecs. A preemphasized 14 pole LPC analysis (autocorrelation method) over 0-5 KHz is done for each window, and the roots of the polynomial are extracted by a pole solving routine [7]. Of the complex conjugate pairs, three pairs with frequency less than 3100 Hz are selected by a formant extraction routine. This procedure reduces the spectral model to a six-pole model defined from 0-3100 Hz. The actual predictor coefficients are then computed by multiplying the three complex conjugate pairs together and collecting terms assuming a sampling frequency of 6200 Hz. All of this preprocessing is computationally expensive but it need be done only once for the entire utterance and requires no extra computation beyond that already necessary for the initial acoustic analysis of our speech understanding system.

The synthesis-by-rule program computes a new set of synthesizer control parameters for every 10 msecs of speech. From these parameters, we derive an all-pole model defining the synthetic spectral shape for each frame of the synthesis. At present, we multiply together the pole pairs defined by the first three formant frequencies and bandwidths for a sampling frequency of 6200 Hz. We then shape the spectrum according to the source excitation generated for that frame.

Given both the measured spectra of an unknown utterance and the synthesized spectra of a hypothesized word, it remains to define a time registration between them. We use the same error metric to do both time normalization and computation of the spectral distance measure. Computing the time normalization means finding the frame-by-frame correspondence between the synthetic and real speech parameterizations. The sequence of these correspondences defines the optimal registration $W(n)$ as shown in Figure 1. The value of $W(n)$ is the frame number of the real speech which corresponds to frame number n of the synthetic parameterization. This means that for every frame of the synthetic parameterization, there is a frame in the utterance which corresponds to it. The reverse is not true, however. There are two sets of constraints which govern computation of the optimal path (Figure 1). These are the boundary conditions and the continuity constraints. In terms of $W(n)$, the boundary conditions are:

$$M-\Delta T_1 \le W(1) \le M+\Delta T_1 \qquad \Delta T_1=\Delta T_2=5$$

$$M'-\Delta T_2 \le W(N) \le M'+\Delta T_2$$

while the continuity constraints are defined as:

$$\begin{aligned} W(n+1)-W(n) &= 0,1,2 \quad (W(n)\neq W(n-1)) \\ &= 1,2 \quad (W(n)=W(n-1)) \end{aligned}$$

Note the continuity constraints imply that the relative durations of the synthetic and real-speech parameterizations must lie between 1/2 and 2. The parallelogram represents the space of all possible time registrations satisfying both sets of constraints.

Itakura [5] provides a detailed description of the computational procedure for finding the optimal registration. Our procedure is identical to his with one important exception. The endpoints of the registration are made variable to allow for uncertainty in the exact position of a given word. Within limits, the DP is allowed to find its own optimal alignment. Because of the configuration, all possible time normalizations as computed by the DP are of equal length in terms of the number of frames matched. The program must examine only those paths terminating at the top of the parallelogram to determine which one is optimal (i.e., has the smallest distance).

4. An Example

Within the scope of our speech understanding system, word matching is done at the phonetic level by the lexical retrieval component. An example of verifying a word which has been proposed by lexical retrieval is given below using the utterance "Give me a list of the remaining trips and their estimated costs" (Figure 2).

In this example, the phonetic transcription of the word "remain", including its lexical stress, is sent to the verification component. Also transmitted are the frame numbers delimiting the portion of the utterance over which the match is to occur. There may be contextual information in the form of an additional phone at one or both ends of the phonetic input string to be verified. This context is used in computing the synthetic parameterization but does not enter into the matching process. Here, no context was given so the program inserted silence ("SIL") at both ends automatically. The duration of [illegible] synthetic template is shown as the ordinate while the unknown utterance extends along the abscissa from the origin to the small vertical mark above the line. The two marks below the line locate the hypothesized alignment (frame numbers) as provided by lexical retrieval. The parallelogram is open slightly at both ends allowing the DP procedure limited freedom in selecting the end points of the optimal registration. For this case, the allowable variations are plus or minus 50 msecs although the one or both may be changed under program control.

The computed spectral distance is normalized by duration and subtracted from 1000. An ideal match having zero distance would have 1000 as its verified match score. In Figure 2, the match score is 724, which is a good score that indicates a likely match, where "remain" was matched against "remain"-ing. This score together with the frame numbers delimiting the optimal registration are transmitted back to the lexical retrieval component where they augment the score already computed for the phonetic word match.

5. Discussion

Although the synthesis-by-rule and matching programs are undergoing further development, we have made some general observations based on our experience with cases like the one above. Where the initial segmentation of an utterance is correct, a word's verification score is approximately the same as its phonetic match score. This agreement holds whether or not the word is correct. In some cases where segmentation or labeling errors have resulted in a prospective word receiving a low phonetic match score, verification was able to offset the effects of these errors by giving a good score to the correct word. The ability to verify a sequence of contiguous words has also been useful since a segmentation error can cause lexical retrieval to overlap the highest scoring occurrences of two words and thereby prevent their being matched sequentially. We are now studying how best to integrate these capabilities into control strategies being developed for the overall speech understanding system.

We are making an effort to improve the performance of the verification component in several respects. The current implementation requires 5-7 times real-time to perform the synthesis and matching procedures on a given word. This could be reduced by introducing a dynamic error bound which terminates execution of the matching algorithm if the distance measure exceeds a certain threshold. As the predictive capabilities of the synthesis-by-rule program improve, we are studying ways to integrate additional parameters into the framework of the existing error metric. One possibility is to use a separate distance measure to do time normalization. This would introduce another parameter into the overall matching process without increasing the dimensionality of the dynamic programming algorithm.

6. Conclusion

We have designed and implemented a verification component for a speech understanding system. The component generates synthetic parameterizations for hypothesized words and matches them onto an unknown utterance. Preliminary results indicate the usefulness of this technique in verifying word hypotheses in continuous speech.

References

[1] Klatt, D.H. and K.N. Stevens (1971) "Strategies for Recognition of Spoken Sentences from Visual Examination of Spectrograms," Bolt Beranek and Newman Report No. 2514, Cambridge Mass.

[2] Klatt, D.H. (1975) "Structure of a Phonological Rule Component for a Synthesis-by-Ruie Program," presented at the 90th Meeting of the Acoustical Society of America, 3-7.

[3] White, G. (1975)
"Automatic Speech Recognition: Linear Predictive Residual versus Bandpass Filtering," Proceeding of the IEEE International Conference on Cybernetics and Society, September 1975.

[4] Bridle, J. and M. Brown (1974)
"An Experimental Automatic Word-Recognition System," Joint Speech Research Unit Report No. 1003, Ruislip, Middlesex, U.K., December 1974.

[5] Itakura, F. (1975)
"Minimum Prediction Residual Principle Applied to Speech Recognition," IEEE Trans. ASSP, 67-72.

[6] Makhoul, J. (1975)
"Linear Prediction in Automatic Speech Recognition," in <u>Speech Recognition</u> (D.R. Reddy, Ed.), New York: Academic Press.

[7] Makhoul, J. and J. Wolf (1973)
"Linear Prediction and the Spectral Analysis of Speech," Bolt Beranek and Newman Report No. 2304, Cambridge Mass.

[8] Woods, W.A. et al. (1975)
"Speech Understanding Systems, Annual Technical Progress Report," Bolt Beranek and Newman Report No. 3188, Cambridge Mass.

[9] Gillmann, R. (1974)
"Automatic Verification of Hypothesized Phonemic Strings in Continuous Speech," Report TM-5-315, System Development Corporation.

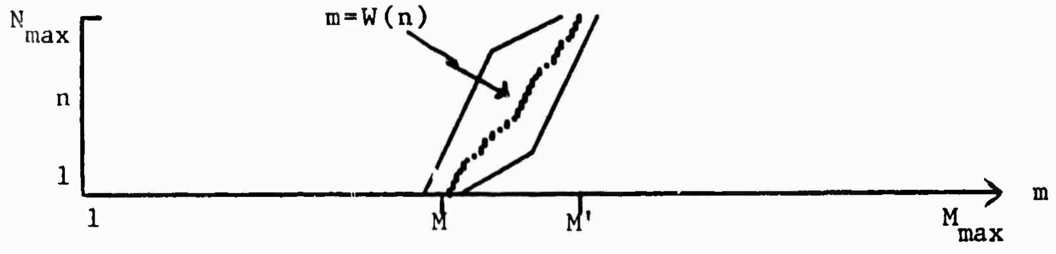

Figure 1

```
 SIL * R IY M 1 EY N * SIL .
Matched  : Left Boundary Time = 99     Right Boundary Time = 137
Verified : Left Boundary Time = 101    Right Boundary Time = 137
The Verified Score is 72ᵘ
```

Figure 2

F. Uses of Higher Level Knowledge in a Speech Understanding System:

A Progress Report

William A. Woods
Madeleine Bates
Geoffrey Brown
Bertram Bruce
John W. Klovstad
Bonnie Nash-Webber

1. Introduction

For several years, the BBN speech understanding project has been developing an experimental speech understanding system for answering questions and responding to commands expressed in continuous speech [1,5,6,7,10,11]. The system is designed to use information from higher knowledge sources such as syntax, semantics, and pragmatics together with sophisticated signal processing and acoustic-phonetic analysis to determine the content of a speech signal. From an initial task application of lunar geology [9], we have recently shifted to that of a travel budget manager's assistant which keeps track of planned and taken trips, amounts allocated or spent for them, various budgets and accounts to be charged, fares to frequent destinations, etc. This paper will describe the state of the current system and illustrate its operation with an example.

The system is implemented as a set of nine interacting processes, or forks, under the TENEX time-sharing system on a PDP-10 computer with part of the system written in LISP, part in FORTRAN, and part in BCPL. The individual components, however, are still far from complete and are continually being refined, improved, and extended. We are using the current system to explore alternative control strategies, to test capabilities of the individual components functioning in a total system, and to uncover needs for new or extended capabilities.

2. Data Objects

(a) Segment Lattices

The major data structure on which the higher level components of the system operate is a lattice of alternative segmentations of the input speech signal produced by the acoustic-phonetic recognition (APR) component [7,8]. This structure specifies a vector of probabilities for each of the possible phonetic identities of each lattice segment. The use of a segment lattice as output from the APR component permits the higher level components to operate with a complete inventory of all of the possible alternative acoustic interpretations which could be placed on the input signal, together with the relative likelihood of each being correct.

(b) Theories

Sentence recognition in the system is based on the manipulation of a data type called a "theory" which represents a partial hypothesis about the identity of the unknown utterance. Theories can range from simple ones, postulating one word that might be in the utterance, to complete ones specifying a sequence of words which covers the utterance, with associated syntactic structure(s) and semantic interpretation(s). Theories may in general contain several distinct contiguous word sequences, called "islands," although the current recognition strategy uses only single island theories. Each theory carries with it a variety of scores assigned by different knowledge sources, together with an overall score which the speech understanding controller uses to determine which theories to pursue.

(c) Monitors, Notices, and Events

When examining a theory, a component may indicate that the theory could be extended by the addition of a word or phrase by setting a data object called a "monitor" on the desired item. A monitor will recall a theory for further extension if an element satisfying the monitor is subsequently found. For example, Syntax can set a monitor for a particular word, syntactic category, or type of constituent, beginning or ending at a particular point in the utterance, and Semantics can monitor for a given semantic category anywhere in the utterance.

When a monitor is satisfied at any point in the understanding process, a "notice" of that fact is created, which results in a pending event being placed on an event queue for further action by the Speech Controller. Events carry scores which enable the Speech Controller to rank order them in order to pursue the most likely ones first.

3. Higher Level Knowledge Sources

(a) Syntactic/Semantic/Pragmatic Support

The current strategy in the BBN speech understanding system relies heavily on a pragmatic grammar that characterizes not just syntactically correct sentences, but also requires sentences to be semantically meaningful and pragmatically appropriate. Such a grammar is specified in terms of semantic categories like trip, person, and expense, rather than purely syntactic categories like noun, verb, and adjective. Although such a grammar is highly topic specific and would have to be rewritten for other applications, it provides a simple organizing principle for applying syntactic, semantic, and pragmatic constraints to the predictions made in the course of the understanding process. For the remainder of this paper, we will refer to all such predictions as "syntactic', but the reader should be aware that they include the use of semantic and pragmatic information.

(b) Lexical Retrieval

The lexical retrieval component of the system consists of a highly refined program for determining the best n matching words in the entire vocabulary or a subset defined by a list of acceptable words and categories for any region of the utterance. The program makes use of a distributed key representation of the dictionary which merges common parts of different words, making an effective search of the entire dictionary computationally feasible without having to consider each word separately. The program provides for matching words both left-to-right and right-to-left and provides a very effective technique for dealing with across-word-boundary phonological effects [4]. Because of the efficiency of this component, it is reasonable for the syntactic component to predict all of the possible words and categories that are acceptable adjacent to a given island.

(c) Control Strategy

The BBN speech understanding system is set up to explore a variety of different control strategies for understanding speech. In this section, we will describe one particular control strategy, which we will call "island-driven." Its operation is as follows:

After a signal has been acquired and the front-end processing has been done to produce a segment lattice, the Speech Controller calls lexical retrieval to scan the entire lattice to determine the (approximately) 15 best matching words. In order to deal with across-word-boundary effects at both ends of each word, this process is done both left-to-right and right-to-left. The words that come back from this scan, as well as all words resulting from later scans, are stored in a "word lattice", where they are available to be noticed and used by later processes. Control then makes a one-word theory for the best matching word which it will try to extend to a complete theory. If the best word fails to produce a complete theory, then the next best word is tried, and so on until either a fixed limit of effort is exceeded or all such theories are exhausted, in which case the system has failed to understand the utterance. When several of these "best matches" score equally, they are processed in parallel.

In order to extend a theory, the Controller calls the syntactic component to evaluate the theory and make predictions for adjacent words to extend it. The syntactic component notices any words that may already be in the word lattice that it can use and sets monitors for other acceptable words and categories. It also makes proposals that can cause the lexical retrieval component to look for specific words or categories. In the island-driven strategy, all acceptable words and categories at the current position are proposed.

After the syntactic component has been called to make its predictions, the Controller is left with a set of proposed words and categories and a set of events resulting from notices (which it can process to extend a theory). At this point, the Controller calls Lexical Retrieval to check the proposals in order to obtain a complete set of events, and then works on the best event. (Events are given a score which is approximately the sum of the lexical score of the theory and the lexical score of the word being used to extend it. Lexical scores, theory scores, and event scores are represented as scaled log probability ratios.)

4. An Example

To further clarify the operation of the island-driven strategy, we will present an example of its performance illustrated by fragments of a computer trace. The sentence is "What is the total budget figure." A trace of the processing up through the initial scan is:

```
17_SYNTAX-DRIVEN(JJW102 I)
DICTIONARY = <KLOVSTAD>RULESDICTSTRUCT.TRAVELDICT-!16
MATCHER = <KLOVSTAD>NEWMATCH.SAV
"31-DEC-75 00:31:03"
Using sentence: JJW102
```

```
RIGHT-TO-LEFT
1 FIGURE 17 24 182 0 -- R
2 FIGURE 17 22 178 -38 -- R
3 TOTAL 7 11 174 -10 -- R
4 FIGURE 17 23 169 -53 -- R
5 YEAR 20 23 107 -23 -- R
6 YOU 20 22 100 -31 -- R
7 IS 3 5 96 -31 -- R
8 ABOVE 10 14 94 0 -- R
9 BUDGET 11 17 81 -16 -- R
10 IT 6 8 80 -16 -- R
11 HIS 2 5 76 -31 -- R
12 TO 7 9 73 -46 -- R
13 WOULD 0 3 72 -31 -- R
14 -S 4 5 72 0 -- R
15 FIGURE 17 21 69 -38 -- R
LEFT-TO-RIGHT
16 TOTAL-ED 7 12 197 -10 -- L
17 FIGURE 17 24 182 0 -- L
18 WHAT 0 3 178 0 -- L
19 FIGURE 17 22 178 -38 -- L
20 TOTAL 7 11 174 -10 -- L
21 FIGURE 17 23 169 -53 -- L
22 BUDGET 11 17 154 -16 -- L
23 YEAR 20 23 107 -23 -- L
24 YOU 20 22 100 -31 -- L
25 IS 3 5 96 -31 -- L
26 FIGURE-ED 17 23 89 -38 -- L
27 FIGURE 17 22 83 0 -- L
28 BUDGET 11 17 81 -16 -- L
29 IT 6 8 80 -16 -- L
30 HIS 2 5 76 -31 -- L
```

Here the format of a word match is a list consisting of a unique word match number, the word, the positions of the left and right end points, the lexical score, the log pronunciation likelihood associated with the word, a list of special features associated with the pronunciation (here all NIL), and an indication whether the word match was left-to-right (L) or right-to-left (R).

The best matching word is TOTAL-ED (the computer name for the regularly inflected word "totaled"). This match extends from position 7 to position 12 in the lattice with a score of 197. A one-word theory for this word is constructed and given to Syntax for evaluation and prediction. It happens that the pragmatic grammar does not permit any utterances with this word in the past tense (although there was no way for the dictionary expansion component which created the inflected form to know that). Consequently no predictions are made for this theory and control turns to the next best matching word, FIGURE.

Notice that there are seven different matches for FIGURE at approximately the same place with various different scores. This type of multiple match for a word is a frequent occurrence due to different possible phonological effects at the ends of a word, different possible segmentations in the segment lattice, and different possible pronunciations of a word. In this case, the multiple matches are due to the uncertainty of segmentation in the last phoneme in the utterance. To avoid redundant processing of such matches, we have introduced the concept of a "fuzzy word match," which is a collection of matches for the same word in approximately the same place. Whenever a word match can be included in such a fuzzed match, the fuzzy word match is used. In this case, a one-word theory (theory 2) for the fuzzy word match of FIGURE is created and given to Syntax.

In processing theory 2, Syntax notices the word BUDGET ending at 17 and makes no other proposals. (In the pragmatic grammar, "figure" is only used in the phrase "budget figure," so no other words are possible on the left; and, since the word is at the end of the utterance, no proposals are possible on the right.) As a result of this notice, an event is created that the Controller now uses to create the theory (BUDGET FIGURE) from 11 to the end of the sentence. Processing this theory results in the trace:

Noticing:

```
WORD THEORY#3 STARTING
  3 TOTAL 7 11 174 -10 -- R
Proposals now are:
1  LIST AND OF PRINT OUT BOTH ME THE A ACTUAL ESTIMATE-V-ED
   POSS
     LEFT/OF
              9 BUDGET 11 17 81 -16 -- R
SOURCE: SYN
About to do proposals:
33 OF 10 11 4 -16 -- R
34 A 10 11 4 -16 -- R
35 THE 9 11 -105 -16 -- R
36 THE 9 11 -105 -16 -- R
37 OUR 10 11 -123 -31 -- R
38 THE 9 11 -135 -16 -- R
39 -S 10 11 -140 0 -- R
40 AND 9 11 -163 -26 -- R
41 OUR 9 11 -168 -46 -- R
42 ME 9 11 -189 -46 -- R
```

The word matches 33 through 42 are the best matching words above a threshold of -200 in lexical score which satisfy the syntactic proposals. Monitors previously set by Syntax notice words in this list and create events for them. As a result, there are 8 events in the event queue waiting to be processed with scores ranging from 442 to 81. The best event is the one linking TOTAL to theory 3, and the Controller decides to follow this one, resulting in the trace:

```
Doing first event:

Creating THEORY#4
  3 TOTAL 7 11 174 -10 -- R
  9 BUDGET 11 17 81 -16 -- R
  1 FIGURE 17 24 182 0 -- R
  17 FIGURE 17 24 182 0 -- L
  19 FIGURE 17 22 178 -38 -- L
  2 FIGURE 17 22 178 -38 -- R
  21 FIGURE 17 23 169 -53 -- L
  4 FIGURE 17 23 169 -53 -- R
  27 FIGURE 17 22 83 0 -- L
  (THEORY#3)
Proposals now are:
1  LIST AND OF PRINT OUT BOTH ME THE A
   POSS
     LEFT/OF
                 3 TOTAL 7 11 174 -10 -- R
SOURCE: SYN
```

(The display of the theory here includes all of the word matches, including the individual matches in fuzzy word matches, plus the additional information about the theory that its parent is theory 3.)

As a result of doing these proposals, new word matches are found and events are created for linking new words into the current theory. The best such is an event linking the word THE to theory 4 even though the lexical score of the best of four fuzzy matches of THE is only 8. This is typical behavior for small function words since they are usually unstressed and frequently poorly pronounced. This event results in the creation of theory 5 (THE TOTAL BUDGET FIGURE), for which Syntax notices the words "is" and "-s" (the contracted verb) in the word lattice (found in the initial scan). Syntax also proposes the words WERE, ARE, OF, and ESTIMATE-V (the verb pronunciation of the word "estimate"). Of the proposals, the lexical retrieval component finds several matches of the words ARE and OF with scores below -100. The best event is thus the notice linking IS (score 96) to theory 5 to produce theory 6 (IS THE TOTAL BUDGET FIGURE).

In processing theory 6, Syntax proposes the words HOW@MUCH and WHAT to the left, and lexical retrieval finds WHAT starting at position 0 with a score of 178 to produce the total theory 7 (WHAT IS THE TOTAL BUDGET FIGURE). Syntax accepts this as a complete utterance and gives it the syntactic structure:

```
S Q
    SUBJ NP DET THE
            ADJ TOTAL
            ADJ BUDGET
            N FIGURE
            FEATS NU SG
    AUX TNS PRESENT
    VP V BE
       OBJ NP D
              N WHAT
              FEATS NU SG/PL
```

5. Conclusions

The above strategy is only one of many that can be implemented with rather minor modifications of the current programs. In particular, a left-to-right or a right-to-left strategy would result from a decision to consider words only at the left end or the right end of the sentence as possible anchors for theories. Such a one-directional strategy would avoid the cost of the initial scan (not a significant fraction of the total time), but would pay the price of anchoring a theory on a word which stands a greater chance of being wrong. (Unusual phonological events happen at the beginnings and ends of sentences increasing the chance that the correct word will not be found there or will be scored very low.) In general, strategies which rely heavily on predictions of next words pay a price when the theory on which those predictions are based is incorrect. The island-driven strategy has the advantage of starting with the acoustically best matching word, wherever it happens to occur in the utterance.

Other control strategies which we are exploring involve the use of a parametric-level word verification component [3], non-adjacent semantic monitors and notices [5], discourse-level predictions of the content of an utterance [2], use of stress and other prosodic information, and varying degrees of reliance on context dependent proposals.

We have tried here to give some flavor of the structure and operation of the current system and the kinds of problems that we are working on. The system has now reached the point where we can begin to run interesting experiments using it, and we expect to learn much during the coming year.

References

[1] Bates, M. (1975)
"The Use of Syntax in a Speech Understanding System." IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-23, No. 1, pp. 112-117.

[2] Bruce, B. (1975)
"Discourse Models and Language Comprehension," American Journal of Computational Linguistics, Microfiche 35, pp. 19-35.

[3] Cook, C. (1976)
"Word Verification in a Speech Understanding System," IEEE International Conference on Acoustics, Speech and Signal Processing, Philadelphia, April, 1976.

[4] Klovstad, J. (1976)
"Probabilistic Lexical Retrieval Component with Embedded Phonological Word Boundary Rules," (forthcoming).

[5] Nash-Webber, B. (1975)
"Semantic Support for a Speech Understanding System," IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-23, No. 1 pp. 124-129.

[6] Rovner, P., B. Nash-Webber and W.A. Woods (1975)
"Control Concepts in a Speech Understanding System," IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-23, No. 1,pp. 136-140.

[7] Schwartz, R. and J. Makhoul (1975)
"Where the Phonemes Are: Dealing with Ambiguity in Acoustic-Phonetic Recognition," IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-23, No. 1, pp. 50-53.

[8] Schwartz, R. and V. Zue (1976)
"Acoustic-Phonetic Recognition in BBN SPEECHLIS," IEEE International Conference on Acoustics, Speech and Signal Processing, Philadelphia, April, 1976.

[9] Woods, W.A. (1973)
"Progress in Natural Language Understanding -- An Application to Lunar Geology," AFIPS Proc., 1973 National Computer Conference and Exposition.

[10] Woods, W.A. (1975)
"Motivation and Overview of BBN SPEECHLIS: An Experimental Prototype for Speech Understanding Research," IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-23, No. 1, pp. 2-10.

[11] Woods, W.A. and J. Makhoul (1973)
"Mechanical Inference Problems in Continuous Speech Understanding," Proc. Third International Joint Conference on Artificial Intelligence, pp. 200-207. (Reprinted in <u>Artificial Intelligence</u>, Vol. 5, No. 1, pp. 73-91, Spring 1974).

Official Distribution List
Contract N00014-75-C-0533

Defense Documentation Center
Cameron Station
Alexandria, Virginia 22314

Office of Naval Research
Information Systems Program
Code 437
Arlington, Virginia 22217

Office of Naval Research
Code 1021P
Arlington, Virginia 22217

Office of Naval Research
Branch Office, Boston
495 Summer Street
Boston, Massachusetts 02210

Office of Naval Research
Branch Office, Chicago
536 South Clark Street
Chicago, Illinois 60605

Office of Naval Research
Branch Office, Pasadena
1030 East Green Street
Pasadena, California 91106

New York Area Office
715 Broadway - 5th Floor
New York, New York 10003

Naval Research Laboratory
Technical Information Division
Code 2627
Washington, D.C. 20375

Dr. A. L. Slafkosky
Scientific Advisor
Commandant of the Marine Corps
Code RD-1
Washington, D.C. 20380

Office of Naval Research
Code 455
Arlington, Virginia 22217

cont'd.

Office of Naval Research
Code 458
Arlington, Virginia 22217

Naval Electronics Laboratory Center
Advanced Software Technology Division
Code 5200
San Diego, California 92152

Mr. E. H. Gleissner
Naval Ship Research & Development Center
Computation and Mathematics Department
Bethesda, Maryland 20084

Captain Grace M. Hopper
NAICOM/MIS Planning Branch (OP-916D)
Office of Chief of Naval Operations
Washington, D.C. 20350

Mr. Kin B. Thompson
Technical Director
Information Systems Division (OP-91T)
Office of Chief of Naval Operations
Washington, D.C. 20350

Advanced Research Projects Agency
Information Processing Techniques
1400 Wilson Boulevard
Arlington, Virginia 22209

Commanding Officer
Naval Air Development Center
Warminster, Pennsylvania 18974

Professor Omar Wing
Dept. of Electrical Engineering
& Computer Science
Columbia University in the City of New York
New York, New York 10027

Assistant Chief for Technology
Office of Naval Research
Code 200
Arlington, Virginia 22217